

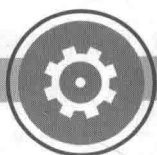
大数据丛书系列之九

总主编◎曾 羽 龙奋杰

机器学习 原理及应用

JIQI XUEXI
YUANLI JI YINGYONG

主 编◎陈海虹 黄 彪 刘 锋 陈文国



电子科技大学出版社

前 言

机器学习是一门多领域的交叉学科,是人工智能的核心,是使计算机具有智能的根本途径,其应用遍及人工智能的各个领域。随着大数据的提出、发展以及大数据时代的到来,机器学习受到了更加广泛的关注,成为行业专家、科研工作者的重要研究内容,同时也受到了很多大学生的青睐。

本书以《关于加快大数据产业发展应用若干政策的意见》和《贵州省大数据产业发展应用规划纲要(2014—2020年)》为指导,以满足贵州省经济社会发展需求为目的,重点介绍了机器学习的基础理论知识和典型方法。本书共有7章,第1章对机器学习的概念、类别、近年来的研究概括和应用前景进行了介绍。第2章阐述了机器学习的基础理论,并以引入实例的方法深入浅出地描述了机器学习的建模过程。第3章重点介绍了神经网络理论基础,并详细讲解了基于神经网络的机器学习规则。第4章对目前主流的机器学习模型分别进行了阐述,包括树模型、概率模型、支持向量机等。第5章对监督学习、集成学习、规则学习、强化学习、流形学习与图谱学习等常见的机器学习方法进行详细讲解。第6章介绍了大数据时代下的数据挖掘与机器学习的相关研究,重点讲述了基于大数据的机器学习应用与研究实例,并对数据挖掘工具进行了简要讲解。第7章列举了机器学习成功应用的实用案例,并进行了详细的解析。

本书由陈海虹担任主编,由黄彪、刘锋、陈文国承担副主编。其中第1章的编写和统稿工作由陈海虹完成,第2、5章的编写工作由黄彪完成,第3、4章的编写工作由刘锋完成,第6、7章的编写工作由陈文国完成。

在编写此书过程中,得到了贵州理工学院曾羽教授、龙奋杰教授的悉心指导,得到了贵州大学李少波教授、杨旭东教授的支持和帮助,贵州理工学院纪斌、聂龙两位老师做了大量工作,在此表示由衷的感谢。此外,编写此书参阅了相关文献资料,借此谨向其作者深表谢忱。由于编者的经验和自身水平有限,难免存在错误或不足之处,恳请读者批评指正。

编 者

目 录

第1章 绪论	1
1.1 什么是机器学习	1
1.2 机器学习的发展历程及研究现状	2
1.2.1 机器学习的发展历程	2
1.2.2 机器学习的研究现状	3
1.3 机器学习的分类	10
1.3.1 基于学习策略的分类	11
1.3.2 基于学习方法的分类	11
1.3.3 基于学习方式的分类	11
1.3.4 基于数据形式的分类	12
1.3.5 基于学习目标的分类	12
1.4 机器学习的应用前景	12
1.4.1 数据分析与挖掘	12
1.4.2 模式识别	13
1.4.3 在生物信息学上的应用	14
1.4.4 更广阔的领域	15
第2章 机器学习基础理论	17
2.1 引言	17
2.2 线性建模	18
2.2.1 定义模型	18
2.2.2 模型假设	18
2.2.3 理想模型的定义	19
2.2.4 最小二乘解	21
2.3 向量和矩阵	23
2.4 线性模型的非线性响应	25

2.5 泛化与过拟合	26
2.5.1 验证数据	26
2.5.2 交叉验证	27
2.5.3 K折交叉验证的计算缩放	27
2.6 噪声	28
2.7 随机变量和概率	29
2.7.1 随机变量	30
2.7.2 概率和概率分布	31
2.7.3 概率的加法	32
2.7.4 条件概率	33
2.7.5 联合概率	34
2.7.6 边缘化	35
2.7.7 贝叶斯规则	37
2.7.8 期望值	38
2.8 常见的离散分布	40
2.8.1 伯努利分布	40
2.8.2 二项分布	40
2.8.3 多项分布	41
2.9 连续型随机变量—概率密度函数	41
2.10 常见的连续概率密度函数	43
2.10.1 均匀密度函数	43
2.10.2 β 密度函数	44
2.10.3 斯密度函数	45
2.10.4 多元高斯	46
2.11 似然估计	48
2.11.1 数据集的似然值	48
2.11.2 最大似然	49
2.11.3 最大似然解的特点	51
2.11.4 最大似然法适用于复杂模型	53

2.12	噪声对参数估计的影响	54
2.12.1	参数估计的影响	54
2.12.2	参数估计的不确定性	55
2.13	预测值的变异性	56
2.13.1	预测值变异性示例	57
2.13.2	估计值的期望值	59
2.14	评估假设	61
2.14.1	动机	61
2.14.2	估计假设精度	62
2.14.3	采样理论基础	65
2.14.4	推导置信区间的一般方法	72
2.14.5	两个假设错误率间的差异	74
2.14.6	学习算法比较	76
第 3 章	人工神经网络	81
3.1	概述	82
3.1.1	什么是人工神经网络	82
3.1.2	神经网络的基本特点与功能	83
3.2	神经网络的应用领域介绍	85
3.3	生物神经网络基础	88
3.3.1	生物神经元的结构	89
3.3.2	生物神经元的信息处理机理	90
3.4	人工神经网络模型及其性能	92
3.4.1	人工神经元模型	92
3.4.2	人工神经网络模型	97
3.5	神经网络学习	105
3.5.1	Hebb 学习规则	107
3.5.2	离散感知器学习规则	108
3.5.3	连续感知器学习规则	109
3.5.4	最小均方学习规则	110

3.5.5 相关学习规则	111
3.5.6 胜者为王学习规则	111
3.5.7 外星学习规则	111
3.6 神经网络系统设计与软硬件实现	112
3.6.1 神经网络系统总体设计	113
3.6.2 神经网络的软件实现	119
3.6.3 神经网络的高级开发环境	120
3.6.4 神经网络的硬件实现	127
第4章 主流的机器学习模型	140
4.1 树模型	140
4.1.1 决策树	143
4.1.2 作为减小方差的树学习方法	144
4.2 概率模型	149
4.2.1 正态分布及其几何意义	150
4.2.2 属性数据的概率模型	151
4.2.3 通过优化条件似然实现鉴别式学习	153
4.3 支持向量机	153
4.3.1 SVM的基本思想	154
4.3.2 非线性支持向量机	159
4.3.3 支持向量机的学习算法	161
4.3.4 支持向量机的应用研究现状	163
第5章 机器学习方法	167
5.1 机器学习的历史	167
5.2 监督学习	168
5.2.1 未标记样本	168
5.2.2 生成式方法	170
5.2.3 阅读材料	172
5.3 集成学习	173
5.3.1 个体与集成	173

5.3.2 Bagging 与随机森林	175
5.4 规则学习	177
5.4.1 基本概念	177
5.4.2 序贯覆盖	179
5.4.3 剪枝优化	181
5.4.4 阅读材料	183
5.5 基于支持向量机的强化学习	184
5.5.1 半监督 SVM	184
5.5.2 核学习	185
5.5.3 SVM	186
5.6 基于 SVM 的强化学习	188
5.6.1 基于 SVM 的 Q 学习结构	188
5.6.2 基于滚动时间窗机制的 SVM	189
5.6.3 算法步骤	191
5.6.4 仿真研究	191
5.7 流形学习与图谱学习	192
5.7.1 流形学习的基本概念	193
5.7.2 流形学习的降维方法分类	193
5.7.3 构建关系矩阵的方法	194
5.8 李群机器学习	200
5.8.1 李群机器学习的基本概念	200
5.8.2 李群机器学习的泛化能力假设公理该公理	201
5.8.3 李群机器学习的学习模型	202
第 6 章 大数据时代的数据挖掘与机器学习	205
6.1 概论	205
6.2 机器学习与数据挖掘	206
6.2.1 无处不在的机器学习与数据挖掘	207
6.2.2 机器学习与数据挖掘大发展历程	209
6.3 数据挖掘工具	214

6.3.1	Weka	214
6.3.2	Java 和 JVM 语音	214
6.3.3	R语言	216
6.3.4	Python	217
6.3.5	SAS	218
6.3.6	数据挖掘工具 Weka 和 R的比较分析	219
6.4	基于数据挖掘和机器学习的木马检测系统	221
6.4.1	网页木马概述	222
6.4.2	网页木马检测系统需求分析	224
6.4.3	网页木马检测系统设计	226
6.4.4	木马检测系统设计小结	242
6.5	机器学习技术在数据挖掘中的商业应用研究	243
6.5.1	技术制造商案例研究	243
6.5.2	在线品牌管理的案例研究	245
6.5.3	移动应用推荐的案例研究	247
第 7 章	机器学习实用案例解析	250
7.1	数据分析	250
7.1.1	分析与验证	250
7.1.2	什么是数据	251
7.1.3	数据推断的类型	253
7.1.4	推断数据的含义	254
7.1.5	数值摘要表	255
7.1.6	均值、中位数、众数	255
7.1.7	分位数	257
7.1.8	标准差和方差	258
7.2	智能收件箱	260
7.2.1	次序未知时该如何排序	260
7.2.2	按优先级给邮件排序	262
7.2.3	文本分析——经典的垃圾邮件过滤系统	262

7.3 搜狗输入法案例解析	265
7.3.1 语义搜索引擎的底层技术和原理	265
7.3.2 顺序分析——搜狗输入法的智能纠错系统	267
7.3.3 搜狗输入法的王牌词库和智能算法	267
7.3.4 频繁树模式和顺序分析算法	268
7.4 基于机器学习的行人检测	270
7.4.1 基于机器学习的行人检测综述	270
7.4.2 基于整体特征的检测方法	271
7.4.3 基于 boostedcascade 的物体检测	276
7.4.4 基于 boostedcascade 的行人检测	281
7.5 机器学习在运动合成与分析中的应用	286
7.5.1 运动合成中的回归与函数逼近	286
7.5.2 降维在运动合成中的应用	288
7.5.3 分类在运动合成中的应用	290
7.5.4 聚类在运动合成中的应用	291
7.5.5 决策在运动合成中的应用	292
参考文献	294

第1章 绪论

1.1 什么是机器学习

机器学习(Machine Learning, ML)作为一门多领域的交叉学科,涉及概率论、统计学、微积分、代数学、算法复杂度理论等多门学科。它通过让计算机自动“学习”的算法来实现人工智能,是人类在人工智能领域展开的积极探索。

在进行此理论学习前,我们应该了解学习是什么。众所周知学习是人们获取外界信息的一种行为,这种行为是后天培养起来的。如何有效地学习,从古至今没有一个权威的答案。各个学派均有不同的观点或见解,到目前为止这个问题仍然存在。究其原因离不开以下三个方面:首先,学习是人的一种主观意识活动,它与人的行为、视觉、认识息息相关,让人们对它的机理难以辨别;其次,学习是各种需求组成的复合体,它是由探索新知识、不断补充已经获取的知识体系、对已有的知识进行创新或改造的一种实践活动,使得人们对学习的本质产生了不同的观点;再者,对学习研究的专家来自于学术界的各个领域,他们对学习的理解有着不同的思考及看法。

在心理学家的眼中,学习是指“人类生产生活中产生的一系列实践经验”;著名科学家 Simon 认为“学习是在进行系统学习中获得的改进方法”;Minsky 则认为“学习是人的一种心理活动”;Mitchell 在著作中阐述学习是“如果某个任务用特殊的性价来衡定的话,那么在完成任务的过程中,伴随着自身能力的上升,那么我们就称之为学习”。到目前为止,与机器学习的相关文献基本上都认为学习是不断积累经验以完善自身的不足。

上述观点虽然不尽相同,但却都包含了知识获取和能力改善这两个主要方面。所谓知识获取是指获得知识、积累经验、发现规律等;所谓能力改善是指改进性能、适应环境、实现自我完善等。在学习过程中,知识获取与能力改善是密切相关的,知识获取是学习的核心,能力改善是学习的结果。所以可以得出:①学习与经验有关;②学习可以改善系统性能;③学习是一个有反馈的信息处理与控制过程。因为经验是在系统与环境的交互过程中产生的,而经验中应该

包含系统输入、响应和效果等信息。因此经验的积累、性能的完善正是通过重复这一过程而实现的。

通过以上分析,我们可以对学习给出如下较为一般的解释:学习是一个有特定目的的知识获取和能力增长过程,其内在行为是获得知识、积累经验、发现规律等,其外部表现是改进性能、适应环境、实现自我完善等。

机器学习又是怎样一门学科呢?机器学习是利用计算机辅助工具来为人类创造价值,机器学习是一个新的领域,它是实现人工智能与生活生产有机结合而兴起的一门学科。如计算机科学(人工智能、理论计算机科学)、数学(概率和数理统计、信息科学、控制理论)、心理学(人类问题求解和记忆模型)、生物学及遗传学(遗传算法、连接主义)、哲学(Occam 剃刀)等。

机器学习有下面几种定义。

(1)机器学习是一门人工智能的科学,该领域的主要研究对象是人工智能,特别是如何在经验学习中改善具体算法的性能。

(2)机器学习是对能通过经验自动改进的计算机算法的研究。

(3)机器学习是用数据或以往的经验,以此优化计算机程序的性能标准。

一种经常引用的英文定义是: A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

在“科普中国”百科科学词条编写与应用工作项目中,机器学习定义为:是一门多领域交叉学科,涉及概率论、统计学、逼近论、凸分析、算法复杂度理论等多门学科。专门研究计算机怎样模拟或实现人类的学习行为,以获取新的知识或技能,重新组织已有的知识结构使之不断改善自身的性能。它是人工智能的核心,是使计算机具有智能的根本途径,其应用遍及人工智能的各个领域,它主要使用归纳、综合,而不是演绎。

1.2 机器学习的发展历程及研究现状

1.2.1 机器学习的发展历程

从 20 世纪 50 年代研究机器学习以来,不同时期的研究途径和目标并不相

同,可以划分为四个阶段。

第一阶段是20世纪50年代中叶到60年代中叶,这个时期主要研究“有无知识的学习”,这类方法主要是研究系统的执行能力。这个时期,主要通过对机器的环境及其相应性能参数的改变来检测系统所反馈的数据,就好比给系统一个程序,通过改变它们的自由空间作用,系统将会受到程序的影响而改变自身的组织,最后这个系统将会选择一个最优的环境生存。在这个时期最具有代表性的研究就是Samuel的下棋程序。但这种机器学习的方法还远远不能满足人类的需要。

第二阶段从20世纪60年代中叶到70年代中叶,这个时期主要研究将各个领域的知识植入到系统里,在本阶段的目的是通过机器模拟人类学习的过程。同时还采用了图结构及其逻辑结构方面的知识进行系统描述,在这一研究阶段,主要是用各种符号来表示机器语言,研究人员在进行实验时意识到学习是一个长期的过程,从这种系统环境中无法学到更加深入的知识,因此研究人员将各专家学者的知识加入到系统里,经过实践证明这种方法取得了一定的成效。在这一阶段具有代表性的工作有Hayes-Roth和Winson的对结构学习系统方法。

第三阶段从20世纪70年代中叶到80年代中叶,称为复兴时期。在此期间,人们从学习单个概念扩展到学习多个概念,探索不同的学习策略和学习方法,且在本阶段已开始把学习系统与各种应用结合起来,并取得很大的成功。同时,专家系统在知识获取方面的需求也极大地刺激了机器学习的研究和发展。在出现第一个专家学习系统之后,示例归纳学习系统成为研究的主流,自动知识获取成为机器学习应用的研究目标。1980年,在美国的卡内基—梅隆(CMU)召开了第一届机器学习国际研讨会,标志着机器学习研究已在全世界兴起。此后,机器学习开始得到了大量的应用。Strategic Analysis and Information System国际杂志连续三期刊登有关机器学习的文章。1984年,由Simon等20多位人工智能专家共同撰文编写的Machine Learning文集第二卷出版,国际性杂志Machine Learning创刊,更加显示出机器学习突飞猛进的发展趋势。这一阶段代表性的工作有Mostow的指导式学习、Lenat的数学概念发现程序、Langley的BACON程序及其改进程序。

第四阶段从 20 世纪 80 年代中叶到现在,是机器学习的最新阶段。这个时期的机器学习具有如下特点。

(1)机器学习已成为新的边缘学科,它综合应用了心理学、生物学、神经生理学、数学、自动化和计算机科学等,形成了机器学习理论基础。

(2)融合了各种学习方法,且形式多样的集成学习系统研究正在兴起。

(3)机器学习与人工智能各种基础问题的统一性观点正在形成。

(4)各种学习方法的应用范围不断扩大,一部分应用研究成果已转化为商品。

(5)与机器学习有关的学术活动空前活跃。

1.2.2 机器学习的研究现状

机器学习是人工智能及模式识别领域的共同研究热点,其理论和方法已被广泛应用于解决工程应用和科学领域的复杂问题。2010 年的图灵奖获得者为哈佛大学的 Leslie valliant 教授,其获奖工作之一是建立了概率近似正确 (Probably Approximate Correct, PAC) 学习理论;2011 年的图灵奖获得者为加州大学洛杉矶分校的 Judea Pearl 教授,其主要贡献为建立了以概率统计为理论基础的人工智能方法。这些研究成果都促进了机器学习的发展和繁荣。

机器学习是研究怎样使用计算机模拟或实现人类学习活动的科学,是人工智能中最具智能特征、最前沿的研究领域之一。自 20 世纪 80 年代以来,机器学习作为实现人工智能的途径,在人工智能界引起了广泛的兴趣,特别是近十几年来,机器学习领域的研究工作发展很快,它已成为人工智能的重要课题之一。机器学习不仅在基于知识的系统中得到应用,而且在自然语言理解、非单调推理、机器视觉、模式识别等许多领域也得到了广泛应用。一个系统是否具有学习能力已成为是否具有“智能”的一个标志。机器学习的研究主要分为两类研究方向:第一类是传统机器学习的研究,该类研究主要是研究学习机制,注重探索、模拟人的学习机制;第二类是大数据环境下机器学习的研究,该类研究主要是研究如何有效利用信息,注重从巨量数据中获取隐藏的、有效的、可理解的知识。

(1)传统机器学习的研究现状

目前,传统机器学习的研究方向主要包括决策树、随机森林、人工神经网络、贝叶斯学习等方面的研究。

决策树是机器学习常见的一种方法。20世纪末期,机器学习研究者 J. Ross Quinlan 将 Shannon 的信息论引入到了决策树算法中,提出了 ID3 算法。1984 年 I. Kononenko、E. Roskar 和 I. Bratko 在 ID3 算法的基础上提出了 ASSISTANT Algorithm,这种算法允许类别的取值之间有交集。同年, A. Hart 提出了 Chi-Squa 统计算法,该算法采用了一种基于属性与类别关联程度的统计量。1984 年 L. Breiman、C. Ttome、R. Olshen 和 J. Freidman 提出了决策树剪枝概念,极大地改善了决策树的性能。1993 年,Quinlan 在 ID3 算法的基础上提出了一种改进算法,即 C4.5 算法。C4.5 算法克服了 ID3 算法属性偏向的问题,增加了对连续属性的处理,通过剪枝,在一定程度上避免了“过度适合”现象。但是该算法将连续属性离散化时,需要遍历该属性的所有值,降低了效率,并且要求训练样本集驻留在内存,不适合处理大规模数据集。2010 年 Xie 提出一种 CART 算法,该算法是描述给定预测向量 X、条件分布变量 Y 的一个灵活方法,目前已经在许多领域得到了应用。CART 算法可以处理无序的数据,采用基尼系数作为测试属性的选择标准。CART 算法生成的决策树精确度较高,但是当其生成的决策树复杂度超过一定程度后,随着复杂度的提高,分类精确度会降低,所以该算法建立的决策树不宜太复杂。2007 年房祥飞表述了一种叫 SLIQ(决策树分类)算法,这种算法的分类精度与其他决策树算法不相上下,但其执行的速度比其他决策树算法快,它对训练样本集的样本数量以及属性的数量没有限制。SLIQ 算法能够处理大规模的训练样本集,具有较好的伸缩性;执行速度快而且能生成较小的二叉决策树。SLIQ 算法允许多个处理器同时处理属性表,从而实现了并行性。但是 SLIQ 算法依然不能摆脱主存容量

①FRIEDMAN B L, O R, STONE C. Classification and Regression Trees[M]. San Francisco Wadsworth, 1984.

②QUINLAN J R. C4.5: programs for maChine learning[M]. San Francisco: Morgan Kauffman, 1993.

的限制。2000年Rajeev RaSto等提出了PUBLIC算法(Pruning and Building Integrated in Classification),该算法是对尚未完全生成的决策树进行剪枝,因而提高了效率。近几年,模糊决策树也得到了蓬勃发展。研究者考虑到属性间的相关性提出了分层回归算法、约束分层归纳算法和功能树算法,这三种算法都是基于多分类器组合的决策树算法,它们对属性间可能存在的相关性进行了部分实验和研究,但是这些研究并没有从总体上阐述属性间的相关性是如何影响决策树性能。此外,还有很多其他的算法,如Zhang. J于2014年提出的一种基于粗糙集的优化算法、Wang. R在2015年提出的基于极端学习树的算法模型等。

随机森林(RF)作为机器学习重要算法之一,是一种利用多个树分类器进行分类和预测的方法。近年来,随机森林算法研究的发展十分迅速,已经在生物信息学、生态学、医学、遗传学、遥感地理学等多领域开展的应用性研究。

①在生物信息学上,Parkhurst等使用RF研究了沙滩细菌密度与其他变量的影响关系。Smith等用RF对细菌追踪数据进行了研究,并和判别分析方法进行了比较。Alonso等利用生物标记寄生虫进行鱼类种群判别。

②在生态学上,Gislason等利用RF方法对土地的覆盖面积进行了研究,并发现RF与其他组合算法相比训练更快。Jan等基于RF和Logistic模型建立了生态水文分布模型,并通过比较发现RF的预测误差小于Logistic模型。

③在医学上,Lee等利用RF技术对助肺CT图像进行肺结节的自动检测,同时还在RF中加入了CAC。

④在遗传学上,Diaz、Uriate等利用RF方法进行基因识别。Chen和Liu利用RF技术进行了蛋白质相互关系的研究。

⑤在遥感地理学上,Pal、Ham、Gislason等都曾利用RF分类器进行了遥感研究。此外,在其他领域RF也得到广泛研究,如Bart把RF应用于客户关系管理中,发现RF的效果都要优于普通线性回归和Logistic模型;Xie等在RF中融合了抽样技术和成本惩罚,并以银行客户数据为例进行了客户流失预测;Coussement等比较了SVM、Logistic模型和RF客户流失预测能力,当且仅当SVM加入最优参数自动选择器后优于Logistic模型,但RF始终优于SVM;Burez等还将加权随机森林应用到客户流失预测中,通过与标准RF的比较发现,加权随机森林有更好的预测效果。

人工神经网络(Artificial Neural Networks, ANN)是一种具有非线性适应性信息处理能力的算法,可克服传统人工智能方法对于直觉,如模式、语音识别、非结构化信息处理方面的缺陷。早在20世纪40年代人工神经网络已经受到关注,并随后得到迅速发展。ANN的研究始于1943年,心理学家 W. McCulloch 和数理逻辑学家 W. Pitts 首先提出了神经元的数学模型,该模型直接影响着这一领域研究的进展。1948年,冯·诺依曼在研究中提出了以简单神经元构成的再生自动机网络结构;20世纪50年代末, F. Rosenblatt 设计制作了“感知机”,它是一种多层的神经网络,这项工作首次把人工神经网络的研究从理论探讨付诸工程实践;与现在的神经网络系统相比,这虽然是一个很简单的模型,但已经具有了如分布式存贮、并行处理、适应性、连续计算、大脑风格等一些神经网络系统的基本性质,这模型引起了广大学者的研究兴趣,人工神经网络的研究进入第一个高峰期。20世纪60年代初期, Widrow 提出了自适应线性元件网络,这是一种连续取值的线性加权求和阈值网络,在此基础上发展了非线性多层自适应网络,这些实际上就是一种 ANN 模型。到1969年, M. Mins 玲等人出版了 *Pereptron* 一书,对感知器模型进行了深入的分析,提出了该模型的问题和不足。他们的论点在很大程度上影响了人工神经网络的发展,使人工神经网络的研究处于低潮。在经过近20年的沉寂后,1982年,美国加州工学院物理学 Hopfield 提出了 Hofield 神经网络模型,引入了“计算能量”概念,给出了网络稳定性判断。1984年,他又提出了连续时间 Hofield 神经网络模型,为神经计算机的研究做了开拓性的工作。Hofield 开创了神经网络用于联想记忆和优化计算的新途径,有力地推动了神经网络的研究,并引起了巨大的反响。人们重新认识到神经网络的威力以及付诸应用的现实性。随即,研究人员围绕着 Hopfield 提出的方法展开了进一步的研究工作,形成了80年代中期以来 ANN 的研究热潮。1985年,又有学者提出了波耳兹曼模型,在学习中采用统计热力学模拟退火技术,保证整个系统趋于全局稳定点。1986年进行认知微观结构的研究,提出了并行分布处理的理论。随后人工神经网络的研究逐步受到了各个发达国家的重视,美国国会通过决议将1990年1月5日开始的十年定为“脑的十年”,国际研究组织号召它的成员国将“脑的十年”变为全球行为。在日本的“真实世界计算(RWC)”项目中,人工智能的研究成了一个

重要的组成部分。人工神经网络与其他传统方法相结合,将推动人工智能和信息处理技术不断发展。近年来,人工神经网络正在模拟人类认知的道路上更加深入发展,与模糊系统、遗传算法、进化机制等结合,形成计算智能,成为人工智能的一个重要方向,将在实际应用中得到发展。将信息几何应用于神经网络的研究,为人工神经网络的理论研究开辟了新的途径。神经计算机的研究发展很快,已有产品进入市场。光电结合的神经计算机为人工神经网络的发展提供了良好条件。人工神经网络的应用已经涉及各个领域,且取得了很大的进展。

贝叶斯学习是机器学习较早的研究方向,其方法最早起源于英国数学家托马斯·贝叶斯在 1763 年所证明的一个关于贝叶斯定理的一个特例。经过多位统计学家的共同努力,贝叶斯统计在 20 世纪 50 年代之后逐步建立起来,成为统计学中一个重要的组成部分。贝叶斯定理因为其对于概率的主观置信程度的独特理解而闻名。此后由于贝叶斯统计在后验推理、参数估计、模型检测、隐变量概率模型等诸多统计机器学习领域方面有广泛而深远的应用。从 1763 年到现在已有 250 多年的历史,这期间贝叶斯统计方法有了长足的进步。在 21 世纪的今天,各种知识融会贯通,贝叶斯机器学习领域有了更广阔的应用场景,发挥了更大的作用。目前,贝叶斯方法特别是最近流行的非参数贝叶斯方法已广泛应用于机器学习的各个领域,并且收到了很好的效果。经典的非参数化贝叶斯方法通常假设数据具有简单的性质,如可交换性或者条件独立等。但是,现实世界中的数据往往具有不同的结构及依赖关系。为了适应不同的需求,发展具有各种依赖特性的随机过程得到了广泛关注。例如,在对文本数据进行主题挖掘时,数据往往来自不同的领域或者类型,我们通常希望所学习的主题具有某种层次结构,为此,层次狄雷克利过程被提出,可以自动学习多层的主题表示,并且自动确定主题的个数。另外,具有多个层次的 IBP 过程也被提出,并用于学习深层置信网络的结构,包括神经元的层数、每层神经元的个数、层间神经元的连接结构等。此外,还有马尔可夫动态依赖关系的无限隐马尔可夫模型、具有空间依赖关系的狄雷克利过程等。另外,对于有监督学习问题,非参数贝叶斯模型最近也受到了广泛的关注。如近期提出的基于 IPB 的非参数化贝叶斯模型,可以自动学习隐含特征,并且确定特征的个数,取得很好的预测性能,

使用 DP 混合模型同时做聚类 and 分类任务也取得了很好的结果。

(2) 大数据环境下机器学习的研究现状

随着大数据时代各行业对数据分析需求的持续增加,通过机器学习高效地获取知识,已逐渐成为当今机器学习技术发展的主要推动力。大数据时代的机器学习更强调“学习本身是手段”,机器学习成为一种支持和服务技术。如何基于机器学习对复杂多样的数据进行深层次的分析,更高效地利用信息成为当前大数据环境下机器学习研究的主要方向。所以,机器学习越来越朝着智能数据分析的方向发展,并已成为智能数据分析技术的一个重要源泉。另外,在大数据时代,随着数据产生速度的持续加快,数据的体量有了前所未有的增长,而需要分析的新的数据种类也在不断涌现,如文本的理解、文本情感的分析、图像的检索和理解、图形和网络数据的分析等。使得大数据机器学习和数据挖掘等智能计算技术在大数据智能化分析处理应用中具有极其重要的作用。在 2014 年 12 月中国计算机学会(CCF)大数据专家委员会上通过数百位大数据相关领域学者和技术专家投票推选出的“2015 年大数据十大热点技术与发展趋势”中,结合机器学习等智能计算技术的大数据分析技术被推选为大数据领域第一大研究热点和发展趋势。

现有许多机器学习算法是基于内存的,大数据却无法装载进计算机内存,故现有的诸多算法不能处理大数据。如何提出新的机器学习算法以适应大数据处理的需求,是大数据时代的研究热点方向。目前,大数据环境下的机器学习研究方向主要包括大数据分治策略、大数据特征选择、大数据分类、大数据聚类、大数据关联分析、大数据的并行研究等。

在大数据的背景下,数据分治策略与抽样作为大数据问题的计算范例,近年取得了较快发展。特别是大部分传统方法,如传统的压缩最近邻(Condensed Nearest Neighbor, CNN)、约减最近邻(Reduced Nearest Neighbor, RNN)、编辑最近邻(Edited Nearest Neighbor, ENN)等只适用于较小规模的数据集,促使大数据的背景下新的数据分治策略与抽样迅速发展,如 Li 等提出了基于局部几何和概率分布来选择分类边缘和边界样本,保持原有数据的空间信息。Angiulli 等在 CNN 的基础上为近邻决策规则提出一种快速压缩最近邻算法(Fast CNN, FCNN),倾向于选择分类边界的样本。Jordan 提出一些关于大

数据的统计推理的方法。当用分治算法来处理统计推理问题时,需从庞大的数据集中获取置信区间。Bootstrap 等通过重新采样数据来获取评估值的波动,进而获取置信区间。数据的不完全抽样会导致错误范围上的波动,必须进行更正以提供校准的统计推理,对此 Jordan 提出一种程序——Bag of Little Bootstraps,可避开这一问题,不仅继承 Bootstrap 的理论性质,并且有许多计算上的优势。分治策略是一种启发式策略,在实际应用中有较好效果,但当试图描述分治算法的统计性能时,新的理论问题就会出现。基于此, Jordan 给出基于随机矩阵浓度定理的理论支持。数据分治与并行处理策略是大数据处理的基本策略,但目前的分治与并行处理策略较少利用大数据的分布知识,且影响大数据处理的负载均衡与计算效率。如何学习大数据的分布知识用于优化负载均衡是一个亟待解决的问题。

大数据特征选择在数据挖掘、文档分类和多媒体索引等新兴领域研究中所面临的数据对象往往是大数据集,其中包含的属性数和记录数都很大,导致处理算法的执行效率低下。大数据环境下,网络流量、通讯记录、大规模社会网络产生大量、多方面的高维数据,所以近年来对这方面的研究较多。如 2006 年 Sagheer 等提出一种快速自组织映射(Fast SOM, FSOM),主要用于解决该问题。该方法的主要思想是:大数据的主要信息主要分布于特征空间的某一区域,如果能找到这些区域并直接在这些区域提取信息,而不是在整个数据空间提取信息,则能大幅度减少时间。2007 年 Quevedo 等基于输入变量的有用性,采用经典技术的简单组合,如相关性和正交性,提出一种输入变量排名算法,用于大数据降维和特征提取,取得良好效果。2009 年 Hua 等对比一些现有的特征选择方法,提出一种特征标签分布式模型,在这个模型中可用特征的数量与它们在真实数据中是一样的,通过它们在一些高维数据上的测试结果显示,不同的特征选择算法在不同的模型条件下性能不一样,它们与样本数量、全局和异构标记的数量有关。2010 年 Gheyas 等结合模拟退火算法、遗传算法、贪心算法及神经网络算法的优点,提出一种模拟退火和遗传算法(Simulated Annealing and Genetic Algorithm, SAGA)混合算法用于解决选择最优化特征子集的 NP 时间问题。2008 年 Kolda 提出一种内存使用高效的 Tucker 分解方法,用于解决传统的张量分解算法无法解决的时间和空间利用问题。该算法在

利用可用内存的前提下最大化计算速度。2010年Pal等提出了一种基于SVM的用于分类的特征选择方法,SVM分类算法的准确度与数据集的特征数和数据集大小有关,因此,在分类前对数据进行特征选择有利于提高分类准确度。2010年Sun等提出一种用于分类的特征选择算法。该算法利用局部学习理论首先将复杂的非线性问题转换为一组线性问题,然后在最大间隔的框架下学习特征关联性。2011年Anaraki等提出了一种带阈值的基于模糊下近似的模糊粗糙集特征选择方法,该方法加入一个阈值来限制Quick Reduct选取特征的数量,与传统的基于模糊下近似的模糊粗糙集特征选择方法(L-FRFS)相比,该方法减少选取特征的数量。实验结果也证明该方法可提高特征提取准确率,减少运行时间。2012年,Wahba探讨了涉及离散、含噪声、不完全相异数据统计/机器学习模型的两种方法:监督、半监督和无监督的机器学习方法。发现这些方法使用训练集中对象之间相异的信息,可得到一个非负的低阶正定矩阵,用于将对象嵌入到一个低维欧几里得空间,其坐标可被用作各种学习模式中的属性。大多数在线学习研究需访问训练实例的所有属性/特征,对高维数据实例或当获得完整的属性/特征集合很昂贵时,这样的经典场景并不总是适合实际应用。为突破此限制,2012年Hoi等通过研究稀疏正则化和截断技术,解决在线特征选择如何利用一个小的和固定数目的活跃特征来准确预测,并提出了一个有效的算法。通过评估所提出算法在一些公共数据集上在线特征选择的经验性能,证明了用在线特征选择技术解决现实世界大数据挖掘问题比一些著名的批处理特征选择算法具有更显著的可扩展性。2013年Song等研究降维技术在轨迹聚类中的作用,运用3种主流降维方法SVD、RP和PCA,结合最通用的轨迹聚类算法,可在计算时间和本地流程模型的适合程度上提高算法性能。由于大数据存在复杂、高维、多变等特性,如何采用降维和特征选择技术以降低大数据处理难度,是大数据特征选择技术迫切需要解决的问题。

在大数据环境下,大数据分类面临的一个新挑战是如何处理大数据。目前大规模数据的分类问题是普遍存在的,但是传统分类算法不能处理大数据。所以对大数据分类的研究得到广泛重视和关注。特别大数据下的支持向量机分类、决策树分类、神经网络与极端学习机、应用领域的分类等方面得到较大发展。

①在大数据下的支持向量机分类方面,2003年Lau等为SVM提出一种在线学习算法,用于处理按顺序逐渐提供输入数据的分类问题。该算法速度更快,所用支持向量个数更少,并具有更优的泛化能力。2006年Laskov等提出了一种快速、数值稳定和鲁棒的增量支持向量机学习方法。2008年Huang等提出了一种大边缘分类器M4,与其他大边缘分类器或局部地或全局地构建分离超平面不同,该模型能局部和全局地学习判定边界。2012年Kim等提出了适用于大数据的特征提取和分类算法。该算法所需内存较少,无须存储较大矩阵,可更好地解决大规模数据分类问题。

②在大数据下的决策树分类方面,2010年Ben-Haim等提出一种构建决策树分类器的算法。该算法在分布式环境中运行,适用于大数据集和流数据,与串行决策树相比,该方法在精度误差近似的前提下能提高效率。2012年Franco-Arcega等提出了一种从大规模数据中构造决策树的方法,解决当前算法中的一些限制条件,可利用所有的训练集数据,但不需将它们都保存在内存中。实验表明,该方法比目前的决策树算法在大规模问题上计算速度更快。2012年Yang等提出了一种增量优化的快速决策树算法用于处理带有噪音的大数据,与传统的挖掘大数据的决策树算法相比,该算法的主要优势是实时挖掘能力,这使得当移动数据流是无限时,它能存储完整的数据用于再训练决策模型。此模型的优点在于在数据流包含有噪音时,能阻止生成决策树大小的爆炸性增长及预测精度的恶化。该模型即使是在含有噪音的数据中,也能生成紧凑的决策树,并有较高的预测精度。

③在神经网络与极端学习机方面,2006年Huang等摒弃梯度下降算法的迭代调整策略,提出了ELM。该方法随机赋值单隐层神经网络的输入权值和偏差项,并通过一步计算即可解析求出网络的输出权值。相比于传统前馈神经网络训练算法需经多次迭代调整才可最终确定网络权值,ELM的训练速度获得较显著提升。

④在应用领域的分类方面,2008年Li等提出了一种半监督的学习算法,用来估计未诊断样本的标记自信度,能较易得出先验知识。该方法在基准数据集上得到较好结果。针对大规模图像数据集的分类性能问题,2011年Lin等提出了在特征提取和分类器训练方面提高效率。对于特征提取,利用Hadoop

架构,在几百个 Mapper 上并行计算。对于训练 SVM,提出并行平均随机梯度下降算法(Parallel Averaging Stochastic Gradient Descent, ASGD),可处理具有 120 万个图像、1000 类的图像数据,具有较快的收敛速度。传统机器学习的分类方法很难直接运用到大数据环境下,不同的分类算法都面临着大数据环境的挑战,针对不同分类算法如何研究并行或改进策略成为大数据环境下分类学习算法研究的主要方向。

大数据聚类是大数据环境下的机器学习重要研究方向之一,随着大数据研究的发展受到越来越多的关注,并得到的快速发展。如 2012 年 Havens 等对比 3 种扩展的模糊 c 均值(FCM)聚类算法对于大数据的执行效率。2012 年 Xue 等提出了一种压缩感知性能提升模型用于大数据聚类,该模型定量分析整个计算过程中与压缩有关的诸多因素的影响。在有上百个计算核的集群上对大到 1.114TB 的 10 维数据进行聚类实验。此外,2009 年 Zhou 提出的基于图的聚类,2011 年 Niu 提出了基于降维的聚类、同年 Vidal R 提出了子空间聚类等。由于经典的聚类算法在大数据环境下面临诸如数据量大、数据体积过大、复杂度高等众多挑战,如何并行或改进现有聚类算法,进而提出新的聚类算法成为研究关键。

大数据并行算法是把传统机器学习算法运用到大数据环境中的重要方法,近几年随着大数据研究的发展产生了很多科研成果。如 2012 年 Luo 等提出了一种非平凡的策略用来并行处理一系列数据挖掘与机器学习问题,由此得到的乘法算法,可直接在如 MapReduce 和通用并行计算架构(Compute Unified Device Architecture, CUDA)的并行计算环境中实现。2012 年 Zhang 等提出了一种大数据挖掘技术,即利用 MapReduce 实现并行的基于粗糙集的知识获取算法,还提出下一步的研究方向,即集中于基于并行技术的粗糙集算法处理非结构化数据。2012 年 Palit 等提出 2 种并行提升算法,即 ADABOOST.PL 和 LOGITBOOST.PL,它们可使多个计算节点同时参与计算并且可构造出一个提升集成分类器,该方法通过利用 MapReduce 框架来实现,在合成数据和真实数据集上的实验表明其在分类准确率、加速比和放大率方面都取得较好结果。2013 年 Kaiser 等利用 MapReduce 框架分布式实现一系列核函数学习机训练,该方法适用于基于核的分类和回归,同时 Kaiser 还介绍一种扩展版的区

域到点建模方法,适应来自空间区域的大量数据。2013年 Upadhyaya 通过图形处理器(Graphic Processing Unit,GPU)平台从并行上得到较显著的性能提升,这些 GPU 平台由于采用并行架构,使用并行编程方法,使得计算能力呈几何级数增长。2016年 Shim 在 MapReduce 框架下,讨论如何设计高效的 MapReduce 算法,对当前一些基于 MapReduce 的机器学习和数据挖掘算法归纳总结,以便进行大数据的分析。并行策略是传统机器学习算法运用于大数据的典型策略之一,并且在一定范围内取得一些进展,能处理一定量级的大数据。如何研究高效的并行策略以高效处理大数据也是当今的研究热点之一。

1.3 机器学习的分类

几十年来,研究发表的机器学习的方法种类很多,根据强调侧面的不同可以有多种分类方法。

1.3.1 基于学习策略的分类

(1) 模拟人脑的机器学习

符号学习:模拟人脑的宏观心理级学习过程,以认知心理学原理为基础,以符号数据为输入,以符号运算为方法,用推理过程在图或状态空间中搜索,学习的目标为概念或规则等。符号学习的典型方法有记忆学习、示例学习、演绎学习、类比学习、解释学习等。

神经网络学习(或连接学习):模拟人脑的微观生理级学习过程,以脑和神经科学原理为基础,以人工神经网络为函数结构模型,以数值数据为输入,以数值运算为方法,用迭代过程在系数向量空间中搜索,学习的目标为函数。典型的连接学习有权值修正学习、拓扑结构学习。

(2) 直接采用数学方法的机器学习

主要有统计机器学习。

1.3.2 基于学习方法的分类

(1) 归纳学习

符号归纳学习:典型的符号归纳学习有示例学习、决策树学习。

函数归纳学习(发现学习):典型的函数归纳学习有神经网络学习、示例学

习、发现学习、统计学习。

(2)演绎学习。

(3)类比学习:典型的类比学习有案例(范例)学习。

(4)分析学习:典型的分析学习有解释学习、宏操作学习。

1.3.3 基于学习方式的分类

(1)监督学习(有导师学习):输入数据中有导师信号,以概率函数、代数函数或人工神经网络为基函数模型,采用迭代计算方法,学习结果为函数。

(2)无监督学习(无导师学习):输入数据中无导师信号,采用聚类方法,学习结果为类别。典型的无导师学习有发现学习、聚类、竞争学习等。

(3)强化学习(增强学习):以环境反馈(奖/惩信号)作为输入,以统计和动态规划技术为指导的一种学习方法。

1.3.4 基于数据形式的分类

(1)结构化学习:以结构化数据为输入,以数值计算或符号推演为方法。典型的结构化学习有神经网络学习、统计学习、决策树学习、规则学习。

(2)非结构化学习:以非结构化数据为输入,典型的非结构化学习有类比学习、案例学习、解释学习、文本挖掘、图像挖掘、Web 挖掘等。

1.3.5 基于学习目标的分类

(1)概念学习:学习的目标和结果为概念,或者说是为了获得概念的学习。典型的概念学习主要有示例学习。

(2)规则学习:学习的目标和结果为规则,或者为了获得规则的学习。典型规则学习主要有决策树学习。

(3)函数学习:学习的目标和结果为函数,或者说是为了获得函数的学习。典型函数学习主要有神经网络学习。

(4)类别学习:学习的目标和结果为对象类,或者说是为了获得类别的学习。典型类别学习主要有聚类分析。

(5)贝叶斯网络学习:学习的目标和结果是贝叶斯网络,或者说是为了获得贝叶斯网络的一种学习。其又可分为结构学习和多数学习。

1.4 机器学习的应用前景

机器学习应用广泛,无论是在军事领域还是民用领域,都有机器学习算法施展的机会,主要包括以下几个方面。

1.4.1 数据分析与挖掘

“数据挖掘”和“数据分析”通常被相提并论,并在许多场合被认为是可以相互替代的术语。关于数据挖掘,现在已有多种文字不同但含义接近的定义,例如“识别出巨量数据中有有效的、新颖的、潜在有用的、最终可理解的模式的非平凡过程”;百度百科将数据分析定义为:“数据分析是指用适当的统计方法对收集来的大量第一手资料和第二手资料进行分析,以求最大化地开发数据资料的功能,发挥数据的作用,它是为了提取有用信息和形成结论而对数据加以详细研究和概括总结的过程。”无论是数据分析还是数据挖掘,都是帮助人们收集、分析数据,使之成为信息,并做出判断,因此可以将这两项合称为“数据分析与挖掘”。

数据分析与挖掘技术是机器学习算法和数据存取技术的结合,利用机器学习提供的统计分析、知识发现等手段分析海量数据,同时利用数据存取机制实现数据的高效读写。机器学习在数据分析与挖掘领域中拥有无可取代的地位,2012年Hadoop进军机器学习领域就是一个很好的例子。

2012年,Cloudera收购Myrrix共创Big Learning,从此,机器学习俱乐部多了一名新会员。Hadoop和便宜的硬件使得大数据分析更加容易,随着硬盘和CPU越来越便宜,以及开源数据库和计算框架的成熟,创业公司甚至个人都可以进行TB级以上的复杂计算。Mydrrix从Apache Mahout项目演变而来,是一个基于机器学习的实时可扩展的集群和推荐系统。

Myrrix创始人Owen在其文章中提到:机器学习已经是一个有几十年历史的领域了,为什么大家现在这么热衷于这项技术?因为大数据环境下,更多的数据使机器学习算法表现得更好,机器学习算法能从数据海洋提取更多有用的信息;Hadoop使收集和分析数据的成本降低,学习的价值提高。Myrrix与Hadoop的结合是机器学习、分布式计算和数据分析与挖掘的联姻,这三大技

术的结合让机器学习应用场景呈爆炸式的增长,这对机器学习来说是一个千载难逢的好机会。

1.4.2 模式识别

模式识别起源于工程领域,而机器学习起源于计算机科学,这两个不同学科的结合带来了模式识别领域的调整和发展。模式识别研究主要集中在两个方面。

(1)研究生物体(包括人)是如何感知对象的,属于认识科学的范畴。

(2)在给定的任务下,如何用计算机实现模式识别的理论和方法,这些是机器学习的长项,也是机器学习研究的内容之一。

模式识别的应用领域广泛,包括计算机视觉、医学图像分析、光学文字识别、自然语言处理语音、识别、手写识别、生物特征识别、文件分类、搜索引擎等,而这些领域也正是机器学习大展身手的舞台,因此模式识别与机器学习的关系越来越密切,以至于国外很多书籍把模式识别与机器学习综合在一本书里讲述。

人脸识别技术的发展就是模式识别发展的一个很好例子。近年来,人脸识别已经有了很多发展。人脸检测技术的提出是人机交互研究发展的需要。人机交互方式,经过第一代的单一文本形式到第二代的图形用户界面的发展,正在向以人为本的方向发展。人们提出了智能人机接口的概念,希望计算机具有或部分具有人的某些智能,人同计算机的交流变得像人与人之间的交流一样轻松自如。

用户是人机界面中的主体,计算机作为一种“智能体”参与了人类的通信活动。人脸检测技术目前已经用于很多领域。在现代社会中,传统的身份鉴定方式(例如口令、信用卡、身份卡等),存在携带不便、容易遗失,或者由于使用过多或不当而损坏、不可读和密码易被破解等诸多问题,已不能很好地满足各种安全需要并显得越来越不适应现代科技的发展和社会的进步。因此,人们希望有一种更加可靠的办法来进行身份鉴定。生物特征识别技术给这一切带来可能。生物特征识别技术(Biometrics)是通过利用个体特有的生理和行为特征来达到身份识别和(或)个体验证目的的一门科学。尽管人们可能会遗忘或丢失他们

的卡片或忘记密码,但是却不可能遗忘或者丢失他们的生物特征如人脸、指纹、虹膜、掌纹等的特征或声音等。

在模式识别技术中,近年来以人脸为特征的识别技术发展十分迅速。相对而言,人脸识别是一种更直接、更方便、更友好、更容易被人们接受的非侵犯性识别方法。作为人脸自动识别系统的第一步,人脸检测技术有着十分重要的作用。

1.4.3 在生物信息学上的应用

随着基因组和其他测序项目的不断发展,生物信息学研究的重点正逐步从积累数据转移到如何解释这些数据。在未来,生物学的新发现将极大地依赖于我们在多个维度和不同尺度下对多样化的数据进行组合和关联的分析能力,而不再仅仅依赖于对传统领域的继续关注。序列数据将与结构和功能数据、基因表达数据、生化反应通路数据、表现型和临床数据等一系列数据相互集成。如此大量的数据,在生物信息的存储、获取、处理、浏览及可视化等方面,都对理论、算法和软件的发展提出了迫切的需求。另外,由于基因组数据本身的复杂性也对理论、算法和软件的发展提出了迫切的需求。而机器学习方法例如神经网络、遗传算法、决策树和支持向量机等正适合于处理这种数据量大、含有噪声并且缺乏统一理论的领域。

机器学习方法在生物信息学中已经有着十分广泛而成功的应用,如 DNA 序列比对、基因及其功能预测、蛋白质结构预测等。神经网络很早就生物序列分析领域获得了应用,1982 年,Stormo 等将以氨基酸序列作为输入向量的感知器用于大肠杆菌核糖体结合位点的预测。1988 年,Qian 等发表了一篇用神经网络模型预测蛋白质二级结构的论文,也使得神经网络得到了足够的重视和真正广泛的应用。1993 年,Borodovsky 等用马尔科夫模型构造了基因发现和基因分析程序 GeneMark,是统计学习理论应用于基因预测领域的一个非常成功的例子。21 世纪以后机器学习在生物信息学中的应用也很多,如 Cheng 等用双聚类方法来分析 DNA 微阵列数据。Long 等用方差分析和贝叶斯统计框架方法来分析大肠杆菌中的基因表达等。

1.4.4 更广阔的领域

目前国外的 IT 巨头正在深入研究和应用机器学习,他们把目标定位于全

面模仿人类大脑,试图创造出拥有人类智慧的机器大脑。

2012年Google在人工智能领域发布了一个划时代的产品——人脑模拟软件,这个软件具备自我学习功能,模拟脑细胞的相互交流,可以通过看YouTube视频学习识别猫、人以及其他事物。当有数据被送达这个神经网络的时候,不同神经元之间的关系就会发生改变。而这也使得神经网络能够得到对某些特定数据的反应机制,据悉,这个网络现在已经学到了一些东西,Google将有望在多个领域使用这一新技术,最先获益的可能是语音识别。

与此同时,Google研制的自动驾驶汽车于2012年5月获得了美国首个自动驾驶车辆许可证。自动驾驶汽车依靠人工智能、视觉计算、雷达、监控装置和全球定位系统协同合作,让电脑可以在没有任何人类主动操作的情况下,通过计算机自动安全地操作机动车辆,Google认为:这将是一种“比人更聪明的”汽车,不仅能预防交通事故,还能节省行驶时间、降低碳排放量。

2013年,Misrosoft CEO高级顾问Craig Mundie在北京航空航天大学学术交流厅发表“科技改变未来”的主题演讲,Mundie在演讲中谈到了当今IT科技的三大挑战:大数据、人工智能和人机互动。他认为随着大数据时代的到来,人们的各种互动、设备、社交网络和传感器正在生成海量的数据,而机器学习可以更好地处理这些数据,挖掘其中的潜在价值。与此同时,他展示了微软研究院在机器学习方面的新产品——英语转汉语实时拟原声翻译,研究过计算语言学的朋友都知道自然语言理解与处理属于机器学习的问题,让计算机理解人类语言可以视同创造出一个机器,该机器拥有与人类一样聪明的智慧。

机器学习在军事上的应用更加广泛,智能无人机、智能无人舰艇、智能无人潜艇陆续研究成功或已投放战场,其他军事领域也有机器学习研究成果的应用。如:美国国防部高级研究计划局的电子战专家正在尝试推出利用机器学习技术对抗敌方的无线自适应通信威胁,其发布了一份概括性机构通告(DARPA-BAA-10-79),内容为“自适应电子战行为学习”计划(BLADE),以研发确保美国电子战系统能够在战场上学习自动干扰新式射频威胁的算法和技术。

第2章 机器学习基础理论

在前面一章中,我们对机器学习概念、发展、分类以及应用前景进行了讨论。我们对机器学习有了初步的了解,知道了机器学习是一门涉及多学科的交叉学科。但在进一步系统的探讨机器学习之前,我们有必要对机器学习的一些基础知识进行了解。本章将通过机器学习的线性建模和一些示例对机器学习基础性理论展开探讨。

2.1 引言

在有着广泛应用的机器学习中,一个重要而且普遍的问题是学习或者推断属性变量与相应的响应变量或目标变量之间的函数关系,它可使得对任何一个属性集合,我们可以预测其响应。

例如,我们可能想要建立一个能够执行疾病诊断的模型。为了构建这个模型,需要使用一个数据集,这个数据集是从已知疾病状态(响应,健康或患病)的患者中得到的测量(属性,如血压、心率、体重等)的集合。

又如,我们希望通过某种模型给顾客提出一些购买建议,这时我们需要建立一个关于某个顾客以前买过物品的描述(属性)和该顾客最终是否喜欢该产品(响应)的模型。这个模型可以帮助我们预测顾客可能喜欢的物品,并因此进行推荐。类似的模型有很多,并涉及许多重要的应用领域。

我们通过一个实际例子来考虑机器学习最直接的学习问题——线性建模:在属性与响应之间学习线性关系。图 2-1 显示了从 1896 年开始,每次奥林匹克运动会(简称奥运会)男子 100 米比赛赢得金牌所需的比赛时间。我们的目标是用这些数据学习一个函数模型,此模型依赖于奥运会举办年份和 100 米获胜时间,并且用这个模型预测将来比赛中的获胜时间。显然,年份并不是影响获胜时间的唯一因素,如果我们认真对待这个预测,可能还需考虑其他因素(如主要参赛者的最近情况)。然而,通过图 2-1 可以看出,年份和获胜时间之间至少存在一个统计关系(它不可能是因果关系——时间的流逝并不是获胜时间下降的直接原因),这个例子足以帮助我们引入和发展线性建模的主要思想。

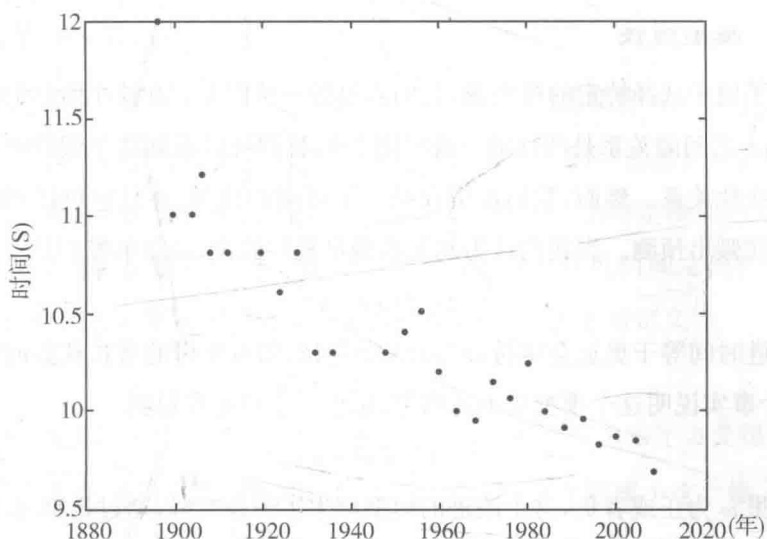


图 2-1 从 1896 年开始,夏季奥运会男子 100 米的获胜时间

(注:在 1914 年、1940 年和 1944 年由于两次世界大战而中断了这个比赛)

2.2 线性建模

2.2.1 定义模型

首先将模型定义为一个输入属性(在上述例子中,属性就是举办奥运会的年份)映射到输出或者目标值(获胜时间)的函数。对于属性,我们用年份的数值(如 1980),尽管还有另外一个公式(例如从第一届运动年开始, $1980 - 1896 = 84$),这对潜在的假设没有实质差别。

有许多函数可以定义这个映射。一般地,这个函数将以 x (奥运会年份)为输入,并且将返回 t (用秒表示的获胜时间)。也就是说, f 是 x 的函数。数学上,把这个记为 $t = f(x)$ 。在有些情况下,我们需要知道的是用来评估函数的 x 。例如,如果 $f(x) = \sin(x)$, 或者 $f(x) = x$, 那么对任何 x , 我们可以计算。一般地,我们需要更灵活的模型,并且我们的模型可能有一个相关参数的集合。例如, $t = ax$ 有一个参数 a , 此参数需要用某种方法定义。在机器学习中,从一个合适的数据集中学习模型参数是一个普遍的问题。我们将用 $t = f(x; a)$ 来表示 x 与参数 a 之间的函数。

2.2.2 模型假设

为了便于选择特定的模型来,我们需要做一些假设。在这个阶段的初始假设, x 与 t 之间的关系是线性的。观察图 2-1,我们可以看到这个假设并不是完全满足线性关系。然而,我们希望它是一个可用的模型,并且它可以对将来的获胜时间做出预测。当我们认为其关系满足我们假设,最简单模型是

$$t = f(x) = x \quad (2-1)$$

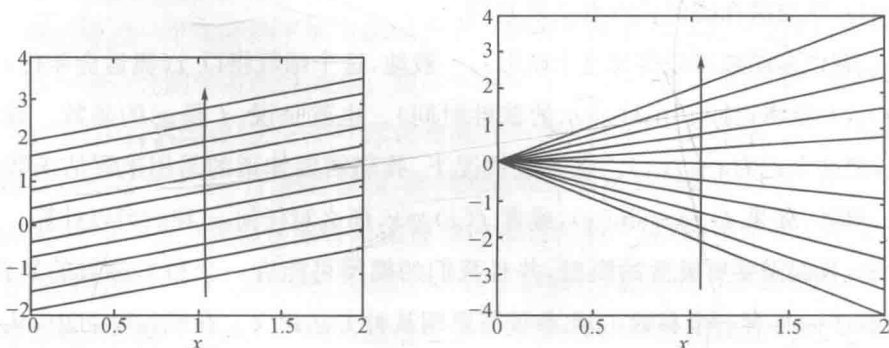
获胜时间等于奥运会年份, $x \geq 1880, t \leq 12$,随着年份的增长获胜时间在下降,这个事实说明这个模型是不适当的,添加一个单参数得到

$$t = f(x; \omega) = \omega x \quad (2-2)$$

这里 ω 为正或者负,这个改进的模型产生了一条直线,通过选择 ω ,可以使这条直线有任何梯度。这个模型在灵活性方面有所提升,但是它仍然是受限制的。因为在奥运会年份 0 年时,模型预测的获胜时间是 $\omega \times 0 = 0$ 。通过这个数据可以看出,这是不现实的,按照数据的一般趋势,在 0 年时,获胜时间实际上应该是一个相当大的数。通过对模型添加多个参数,可以克服这个限制:

$$t = f(x; \omega_0, \omega_1) = \omega_0 + \omega_1 x \quad (2-3)$$

这是直线的标准等式,这个等式许多读者以前都遇到过。现在学习任务是图 2-1 的数据为两个参数 ω_0 和 ω_1 选择合适的值。这两个参数常常认为是截距(ω_0 ,直线与 t 轴的截距)和梯度(ω_1 ,直线的梯度),以及改变它们的影响(effect),如图 2-2 所示。



(a) ω_0 的增加改变了直线与 t 的相交点

(b) ω_1 的增加改变了直线的梯度

图 2-2 线性模型中参数 ω_0 和 ω_1 产生的影响

2.2.3 理想模型的定义

为了选择在某种方式下最好的 ω_0 和 ω_1 值,我们需要定义理想模型的意义是什么。常识表明所谓理想模型,其解是由 ω_0 和 ω_1 的一些值组成,这些值可以产生一条尽可能与所有数据点接近的直线。衡量一个特定模型与数据点接近程度的普遍方法是真正的获胜时间与模型预测的获胜时间之间的平方差。用 x_n, t_n 分别表示第 n 次的奥运会年份和获胜时间,平方差定义为

$$(t_n - f(x_n; \omega_0, \omega_1))^2 \quad (2-4)$$

这个数值越小,模型在 x_n 处越接近 t_n 。在这里对差值取平方是很重要的,如果不这样做,就可以通过连续增加 $f(x_n; \omega_0, \omega_1)$ 来无限减小这个量。

$(t_n - f(x_n; \omega_0, \omega_1))^2$ 称为平方损失函数,因为它描述了使用 $f(x_n; \omega_0, \omega_1)$ 模拟 t_n 所损失的精度。在本章中,我们用 $L_n()$ 表示损失函数,即

$$L_n(t_n, f(x_n; \omega_0, \omega_1)) = (t_n - f(x_n; \omega_0, \omega_1))^2 \quad (2-5)$$

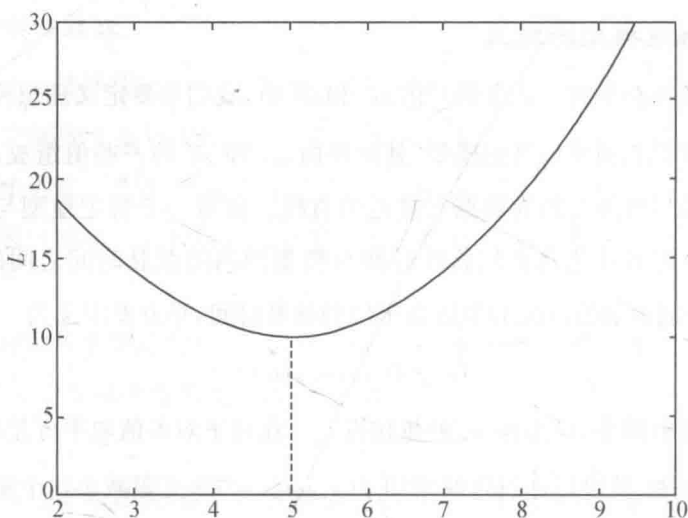
式(2-5)是第 n 年的损失。损失总是正的,并且损失越小,函数描述这个数据就越好。由于对于所有年,我们想有一个低的损失,所以考虑在整个数据集上的平均损失,即

$$L = \frac{1}{N} \sum_{n=1}^N L_n(t_n, f(x_n; \omega_0, \omega_1)) \quad (2-6)$$

这是 N 年的平均损失值,其值越低越好。因此我们将调整 ω_0 和 ω_1 值来产生一个模型,此模型得到平均损失的最低值 L 。寻找 ω_0 和 ω_1 最好值,用数学表达式可以表示为

$$\operatorname{argmin}_{\omega_0, \omega_1} \frac{1}{N} \sum_{n=1}^N L_n(t_n, f(x_n; \omega_0, \omega_1)) \quad (2-7)$$

argmin 项是数学上“找到最小化参数”的缩写。在这个例子中,参数是 ω_0 和 ω_1 的值同时最小化的表达式是平均损失。图 2-3 显示了一个假设的损失,它是单参数 ω 的函数。使 L 最小的参数 ω 的值是 $\omega = 5$ 。历史上,平方损失的最小化是函数估计的最小二乘误差法的基础,它是由 Gauss 和 Legendre(1809 年)在预测行星运动时发展的方法。

图 2-3 单参数(ω)损失函数的例子

(注:虚线表明了 $\omega=5$ 时,损失最小)

除了上述损失函数外,还有其他适合回归的损失函数,例如绝对损失,表达式如下

$$L_n = |t_n - f(x_n; \omega_0, \omega_1)| \quad (2-8)$$

平方损失是非常常见的选择,较大原因是它能相对直接找到 ω_0 和 ω_1 的最好值。然而,现代计算能力已经降低了数学方便的重要性——在多个适合的数据上选择一个方便的损失函数不再困难。然而在本章,我们的目标是介绍对平方损失合适的通用模型概念,对于其他损失函数不做过多阐述。

2.2.4 最小二乘解

基于之前奥运会的函数模型,我们的数据集由 $n=1, 2, \dots, N$ 观测值构成,它们中的每一个由一个年 x_n 和时间(秒) t_n 构成。我们所寻找的函数关系用一个线性模型定义为

$$f(x; \omega_0, \omega_1) = \omega_0 + \omega_1 x \quad (2-9)$$

我们决定将用最小二乘损失函数来选择适合的 ω_0 和 ω_1 ,并用表达式中的线性模型替代平均损失,结果为

$$L = \frac{1}{N} \sum_{n=1}^N (\omega_1^2 x_n^2 + 2\omega_1 x_n (\omega_0 - t_n) + \omega_0^2 - 2\omega_0 t_n + t_n^2) \quad (2-10)$$

对损失函数求导数:在 L 的最小值点处,其关于 ω_0 和 ω_1 的偏导数一定是 0。因此,求出偏导数,利用其值等于 0 求取 ω_0 和 ω_1 。对 ω_1 求偏导时,式(2-10)中不包含 ω_1 的项可以被忽略(由于这些项关于 ω_1 偏导数为 0)。去掉这些项得到

$$\frac{1}{N} \sum_{n=1}^N (\omega_1^2 x_n^2 + 2\omega_1 x_n \omega_0 - 2\omega_1 x_n t_n) \quad (2-11)$$

对 ω_1 求偏导数得

$$\frac{\partial L}{\partial \omega_1} = 2\omega_1 \frac{1}{N} \left(\sum_{n=1}^N x_n^2 \right) + \frac{2}{N} \left(\sum_{n=1}^N x_n (\omega_0 - t_n) \right) \quad (2-12)$$

用同样的方法,对 ω_0 求偏导数得

$$\frac{\partial L}{\partial \omega_0} = 2\omega_0 + 2\omega_1 \frac{1}{N} \left(\sum_{n=1}^N x_n \right) - \frac{2}{N} \left(\sum_{n=1}^N t_n \right) \quad (2-13)$$

导数等于 0:现在我们有损失函数关于 ω_0 和 ω_1 的偏导数表达式。为了找到对应于拐点(希望是最小值点)的 ω_0 和 ω_1 值,必须使这些表达式为 0 并且对 ω_0 和 ω_1 求解。从关于 ω_0 的表达式开始,将式(2-13)设置为 0,并且对 ω_0 求解,通过简化处理后得

$$\omega_0 = \frac{1}{N} \left(\sum_{n=1}^N t_n \right) - \omega_1 \frac{1}{N} \left(\sum_{n=1}^N x_n \right) \quad (2-14)$$

将平均获胜时间表示为 $\bar{t} = \frac{1}{N} \sum_{n=1}^N t_n$, 平均奥运会年份为 $\bar{x} = \frac{1}{N} \sum_{n=1}^N x_n$, 在拐点 $\bar{\omega}_0$ 处。可以重写 ω_0 值的表达式为

$$\bar{\omega}_0 = \bar{t} - \omega_1 \bar{x} \quad (2-15)$$

我们从这个表达式可以洞悉到什么? 这个新的表达式是初始表示($t_n = \omega_0 + \omega_1 x_n$)的重新排列,这里 t_n 和 x_n 已经被平均值 $\bar{t}$ 和 $\bar{x}$ 取代。考虑在 N 个数据点上的平均函数值,表达式如下

$$\frac{1}{N} \sum_{n=1}^N f(x; \omega_0, \omega_1) = \frac{1}{N} \sum_{n=1}^N (\omega_0 + \omega_1 x) = \omega_0 + \omega_1 \bar{x} \quad (2-16)$$

在我们用式(2-12)得到关于 $\bar{\omega}_1$ (ω_1 在拐点处的值)的表达式之前,需简要检验它的 2 阶导数,以确保它是最小点。并需再一次对式(2-12)关于 ω_1 求导,并对式(2-13)关于 ω_0 求导,结果为

$$\frac{\partial^2 L}{\partial \omega_1^2} = \frac{2}{N} \sum_{n=1}^N x_n^2 \quad (2-17)$$

$$\frac{\partial^2 L}{\partial \omega_0^2} = 2 \quad (2-18)$$

这两个量一定都是正的,说明它仅有一个拐点并且此拐点对应于损失函数的最小值。我们将此过程应用于关于 $\bar{\omega}_0$ (最小化损失函数的 ω_0 的值) 的表达式中。这个表达式依赖于 ω_1 , 这暗示了对于特定的 ω_1 , 我们知道最好的 ω_0 。式(2-12)用我们的表达式替代最好的 ω_0 (式(2-15)) 并重新排列, 我们得到仅含 ω_1 项的表达式

$$\frac{\partial L}{\partial \omega_1} = \omega_1 \frac{2}{N} \left(\sum_{n=1}^N x_n^2 \right) + \bar{t} \frac{2}{N} \left(\sum_{n=1}^N x_n \right) - \omega_1 \bar{x} \frac{2}{N} \left(\sum_{n=1}^N x_n \right) - \frac{2}{N} \left(\sum_{n=1}^N x_n t_n \right) \quad (2-19)$$

依然用 $\bar{x} = (1/N) \sum_{n=1}^N x_n$ 来简化这个表达式并合并包含 ω_1 的项

$$\frac{\partial L}{\partial \omega_1} = 2\omega_1 \left[\left(\frac{1}{N} \left(\sum_{n=1}^N x_n^2 \right) \right) - \bar{x}\bar{x} \right] + 2\bar{t}\bar{x} - 2 \frac{1}{N} \left(\sum_{n=1}^N x_n t_n \right) \quad (2-20)$$

最后, 将这个偏导数设置为 0, 我们能得到关于 $\bar{\omega}_1$ 的表达式

$$\bar{\omega}_1 = \frac{\frac{1}{N} \left(\sum_{n=1}^N x_n t_n \right) - \bar{t} \bar{x}}{\frac{1}{N} \left(\sum_{n=1}^N x_n^2 \right) - \bar{x} \bar{x}} \quad (2-21)$$

现在定义一些新的平均量是非常有用和必要的。第一个, $(1/N) \sum_{n=1}^N x_n^2$ 是数据的平均平方值并且我们把它记为 $\bar{x}^2$ 。注意, 这个量与 $(\bar{x})^2$ 不同。第二个是 $(1/N) \sum_{n=1}^N x_n t_n$ (同样, 它与不同)。我们将它记为 $\overline{xt}$ 。将这些在关于 ω_1 的表达式中替换, 得到

$$\bar{\omega}_1 = \frac{\overline{xt} - \bar{x} \bar{t}}{\bar{x}^2 - (\bar{x})^2} \quad (2-22)$$

最后, 可利用式(2-15)和式(2-28)计算出最好的参数值。

2.3 向量和矩阵

在许多应用中, 我们感兴趣的是这样一些问题: 其中每一个数据点表示为一些属性的集合。例如, 我们可以确定, 仅仅用奥运会年份并不适合奥运会短跑数据模型的建立。用奥运会年份和每个运动员个人最好成绩建立的模型可

能更准确。用 $s_1, s_2, \dots, s_8$ 表示在跑道 1-8 的运动员的最好成绩(获胜时间), 合适的线性模型可能包括:

$$t = f(x, s_1, \dots, s_8; \omega_0, \dots, \omega_8) \\ = \omega_0 + \omega_1 x + \omega_2 s_1 + \omega_3 s_2 + \omega_4 s_3 + \omega_5 s_4 + \omega_6 s_5 + \omega_7 s_6 + \omega_8 s_7 + \omega_9 s_8 \quad (2-23)$$

我们可以执行一遍以前的分析来找到 $\bar{\omega}_0, \dots, \bar{\omega}_9$ 。求得损失函数的偏导数之后, 得到 10 个等式, 它们再经过重新排序和相互替换。这是费时的练习, 并且随着包含变量的进一步增加, 它们很快变得不可行(具有数千个变量的机器学习是常见的)。幸运的是, 这里有另一种方法——向量和矩阵。

如果想在向量中表示所有的元素, 我们将它以列的形式写出来, 并用中括号括起来。这里是长度为 2 和 4 的两个向量的例子:

$$x_n = \begin{bmatrix} x_{n1} \\ x_{n2} \end{bmatrix}, y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} \quad (2-24)$$

由于将向量记为列有点笨拙, 所以我们常常将它们记为行, 并且用转置符来表示它们应该被旋转过来。如果我们假设有 D 个属性, 那么将 x_n 定义为 $x_n = [x_{n1}, \dots, x_{nD}]$ 。在我们着手添加额外的变量时, 有必要以向量形式重复对初始模型($t = \omega_0 + \omega_1 x$)的分析。这需要我们对在两种情况下得到的 $\bar{\omega}_0$ 和 $\bar{\omega}_1$ 的表达式进行比较。第一步, 将 ω_0 和 ω_1 合并为单个参数向量 ω , 并将每个 x_n 扩大为 1, 从而产生数据向量 x_n , 即

$$\omega = \begin{bmatrix} \omega_0 \\ \omega_1 \end{bmatrix}, x_n = \begin{bmatrix} 1 \\ x_n \end{bmatrix}$$

依据 x_n 和 ω , 此模型可以表示为

$$f(x_n; \omega_0, \omega_1) = \omega^T x_n = \omega_0 + \omega_1 x_n \quad (2-25)$$

我们可以在任何一个实例用 $\omega^T x_n$ 替换 $\omega_0 + \omega_1 x_n$, 例如, 平方损失 L 可以表示为

$$L = \frac{1}{N} (t - X\omega)^T (t - X\omega) \quad (2-26)$$

在式(2-26)中, X 和 t 为

$$X = \begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_N^T \end{bmatrix} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_N \end{bmatrix}, t = \begin{bmatrix} t_1 \\ t_2 \\ \vdots \\ t_N \end{bmatrix} \quad (2-27)$$

向量和矩阵的微分损失: 我们想要与 L 一个拐点(极小值)一致的向量 ω 的值, 必须求得 L 关于 ω 的偏导数。依次获得 L 关于 ω 每个元素的偏导数, 将结果组成一个向量。尽管在本章后面可以看到, 实际上能够直接获得的向量形式, 但在这个例子中还是值得这样做的。以两个变量为例, 向量表示为

$$\frac{\partial L}{\partial \omega} = \begin{bmatrix} \frac{\partial L}{\partial \omega_0} \\ \frac{\partial L}{\partial \omega_1} \end{bmatrix} \quad (2-28)$$

该向量包含 L 关于 ω_0 和 ω_1 的偏导数。向量的这两个元素依次与式(2-13)和式(2-15)中的元素是一样的。通过运算这两个参数的微分方程式(2-27), 能够检验我们的损失的确是正确的。

2.4 线性模型的非线性响应

这章开始就假设应用线性函数对时间和奥运会 100 米短跑时间之间的关系建立模型。在很多实际应用中, 这是很受约束的。即使对于 100 米数据, 它过于简单化。在对于更复杂的模型, 我们需要通过属性转换正确地使用我们之前描述的框架。

之前我们建立的线性模型 $f(x; \omega) = \omega_0 + \omega_1 x$ 是一个关于参数 (ω) 和 (x) 数据的线性关系式, 从计算的角度看, 参数的线性性可以描述为最小化平方损失函数的解, 如式(2-15)和式(2-22)中。

我们增加列 x_n^2 , 扩展数据短阵 X

$$x_n = \begin{bmatrix} 1 \\ x_n \\ x_n^2 \end{bmatrix}, X = \begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ \vdots & \vdots & \vdots \\ 1 & x_N & x_N^2 \end{bmatrix} \quad (2-29)$$

并且增加一个额外参数 ω :

$$\omega = \begin{bmatrix} \omega_0 \\ \omega_1 \\ \omega_2 \end{bmatrix} \quad (2-30)$$

结果为

$$f(x; \omega) = \omega^T x = \omega_0 + \omega_1 x + \omega_2 x^2 \quad (2-31)$$

由于参数中的模型仍是线性的,所以可以求取 ω ,但是要拟合的函数在数据中是二次的。在图 2-4 中,它就是用数据的二次函数来拟合适当的数据(实线)。从图中可以看到,它通过获得的函数能够拟合原始线性(在数据中)模型(虚线, $t = \omega_0 + \omega_1 x$)。从拟合的结果看,二次模型很明显更适合。

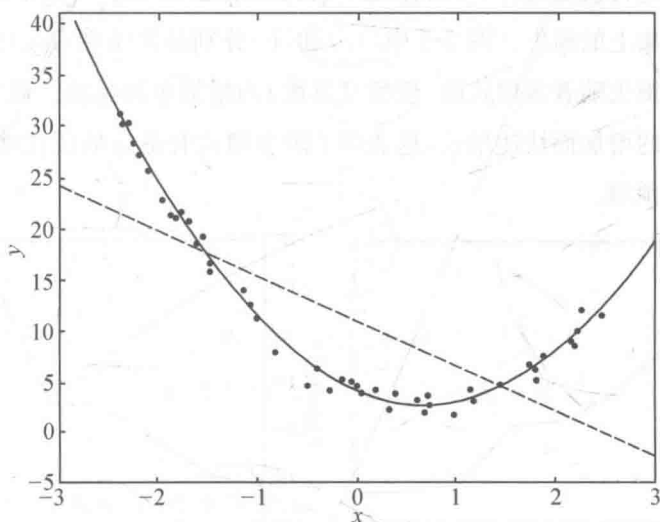


图 2-4 线性和二次模型拟合二次函数生成的数据集

2.5 泛化与过拟合

假定原来建立模型的目的是做预测,那么不难理解最好的模型就是可以预测最精确的模型,即可以泛化训练样本以外数据的模型(例如,到 2008 年的奥运会数据)。理想情况下,我们更喜欢选择在不可见数据上性能最好的模型(即最小化损失),但是由于问题本身的原因,数据无法得到。在之前的模型中,可应用训练数据上的损失选择用于预测的模型,通过比较发现训练数据在 8 阶多

项式拟合数据的损失比 1 阶多项式的更低,但对于未来奥运会的预测表现为非常糟糕。基于 8 阶多项式的模型过于关注训练数据(过拟合),因此不能很好地泛化新数据。由于模型越来越复杂,所以也越来越逼近可观测数据。但是,当超过某点,预测的质量就会迅速退化。为了克服过拟合,能够很好地泛化,确定最优模型的复杂度将会非常有挑战性。

2.5.1 验证数据

克服过拟合问题的一般方法是使用第二个数据集,即用验证集来验证模型的预测性能。验证数据可以单独提供也可从原始训练集中拿出一部分。例如,在 100 米数据中,可以从训练集中拿出 1980 年以后的所有奥运会数据作为验证集。为了进行模型选择,可以在缩小的训练集上训练每一个模型,然后计算它们在验证集上的损失。图 2-5 中,(a)和(b)分别是训练和(log)验证损失的曲线。训练损失随着多项式阶(模型复杂度)的增加单调递减。而验证损失随着多项式阶的增加而快速增长,这表明 4 阶多项式有最好的泛化能力,能够产生最可靠的预测。

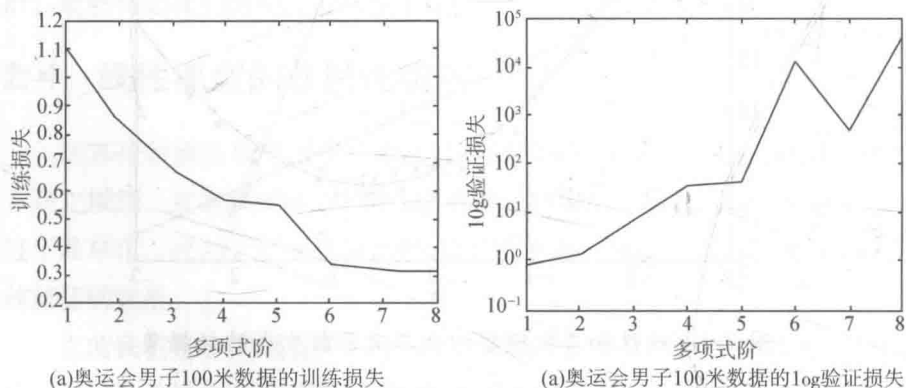


图 2-5 奥运会男子 100 米数据训练和验证损失

2.5.2 交叉验证

根据验证集计算损失对验证集数据选择的敏感性,我们知道如果验证集很小,操作是十分困难的。这时,交叉验证便作为一种有效使用现有数据集的重要方法。

如图 2-6 所示, K 折交叉验证把数据集分成大小相等的 K 份(或者尽可能

相等)。每块轮流作为验证集,其他 $K-1$ 块作为训练集。结果 K 个损失值的平均值作为最后的损失值。 K 折交叉验证的一个极端情况是,当 $K=N$,即 K 恰好等于数据集中的可观测数据的数量时,每个观测数据依次拿出用作测试其他 $N-1$ 个对象训练得到的模型。交叉验证的这种特殊形式称为留一交叉验证(Leave One Out Cross Validation, LOOCV)

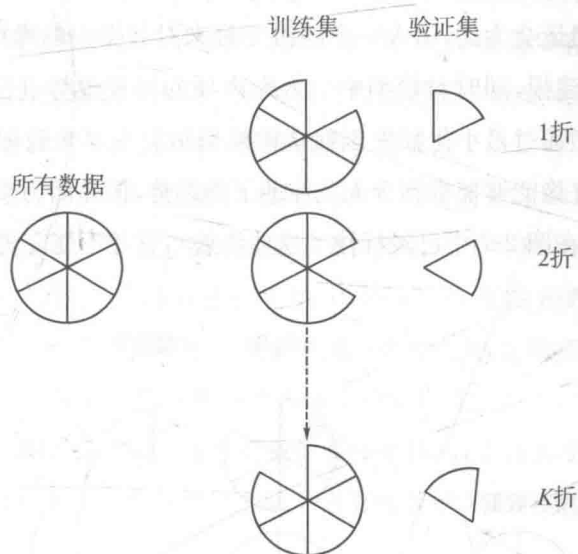


图 2-6 交叉验证

(注:在每一个 K 折,数据点的一个集合从训练集中移出,用于验证或测试模型)

2.5.3 K 折交叉验证的计算缩放

交叉验证似乎是估计训练数据集期望损失的一种好方法,它可以查看和评估各种可选择的模型。然而,考虑 LOOCV 的实现。需要训练模型 N 次,这比只在所有数据上训练一次多耗费大约 N 倍的时间(这样说并不完全准确,因为训练是在小的数据点上进行的)。对于某些模型,尤其是有很多数据的模型,该方法可能并不可行。

缓解这个问题最简单方法就是让 $K \ll N$ 。例如,在 10 折交叉验证中,可以用其中 10% 的数据做验证,其他剩下的 90% 做训练。这样会降低从 $N-10$ 的训练循环数。如果 $N \gg 10$,那么将是一个相当大的节省。通常的选择是用 N

折交叉验证,并且应用不同分割的 N 组数据多次反复,对每次重复的验证结果求平均作为最后的结果。

2.6 噪声

在前面几节介绍了通过定义和最小化损失函数来学习模型参数的方法。本节将继续以奥运会为例,引入一个随机变量来对数据中的噪声(模型和观测值之间的误差)建模,同时对模型中引入噪声项的可观优势进行说明。图 2-7 是使用线性模型通过最小化损失函数来建模奥运会 100 米数据的结果。该线性模型看上去好像能够捕捉到令人关注的下降趋势,但是因为模型和实际数据之间存在误差(在图 2-7 中已经标注了这些误差),它并不能完美地解释每一个数据点。

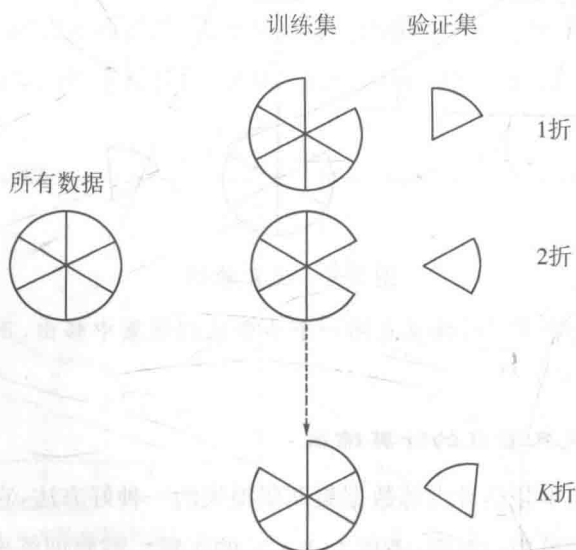


图 2-7 奥运会男子 100 米数据的线性拟合

在构建模型的时候,我们假设年和比赛时间存在线性关系。从图 2-7 可以看出该模型虽然能够捕获数据中的总体趋势,但忽略模型和观测数据之间出现的较大偏差。

从建模的观点来看,很难忽略这些误差。如果我们知道如何表示这些误差,那么我们将它们构建在模型中,这样做对误差建模的确有许多好处。特

别是如果据此适当修改 ω , 这将提高我们在估计模型参数 ω 时表达不确定性的级别, 使我们可在预测中表达不确定性的程度, 即“我们相信获胜时间将在 a 、 b 之间”, 而不是“我们认为获胜时间一定是 c ”。

产生这个特定数据集的过程很复杂的, 因为我们甚至不能构造一个近似完美的模型来表示一个短跑运动员以及影响其准备和表现的事件, 更不用说多个运动员和所有其他因素了。但建模问题当作产生式模型是很有必要的, 我们尝试建立一个模型, 使得这个模型产生与现有数据相似的数据集。尽管我们知道实际数据不是这样产生的, 但是这种策略却非常有效。

我们怎样着手从现在的模型生成数据呢? 对于等式 $f(x; \omega) = \omega^T x$, 如果代入前面得到的 ω 值, 那么这个等式就能够针对每一个特殊的年份产生一个获胜时间。图 2-8 给出了这种方法产生的 1920—2000 年的获胜时间。这与图 2-7 的数据看上去并不是很接近。为了让这组数据更贴近现实, 需要增加一些误差。观察图 2-7 我们发现这些误差有两个重要特征。

(1) 每年的误差都不同。有些是正的有些是负的, 并且大小不同。

(2) 误差的大小或方向与年份之间并没有明显的关系。误差不像是奥运会年份 x 的函数。

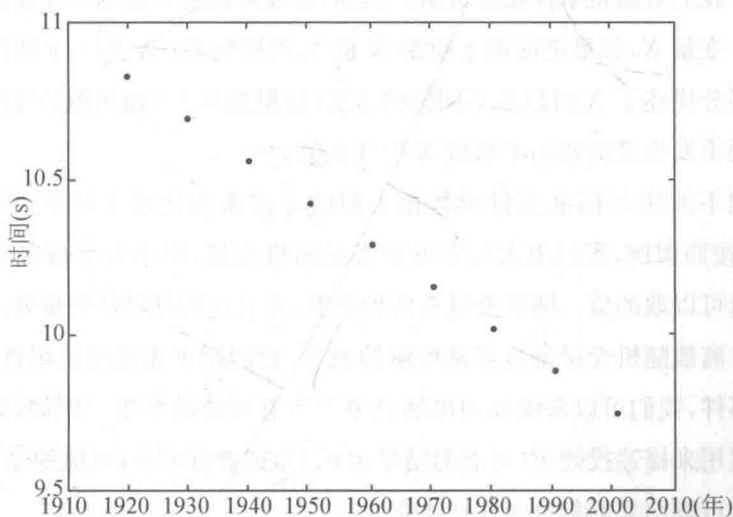


图 2-8 使用线性模型生成的数据集

如果有方法产生或正或负的随机大小的时间(秒级), 并且这个时间与图

2-7的误差大致相等,那么我们就能够针对每一个数据点生成这样一个我们想要的时间,并且可以将它加到 $\omega^T x$ 上。将我们需要的可变性引入模型的工具来源于统计学,相关知识将在下一节进行讲解。

2.7 随机变量和概率

我们建立的任何模型都是产生观测数据的真实系统的简化。这导致了模型和现实的差异。本节介绍的工具将帮助我们建模和理解这种差异。因为我们必须从基础开始,所以刚开始时它可能与将误差加入100米数据和表达预测不确定性这个特定问题无关。但是随着进一步深入,这种联系将逐渐清晰。

2.7.1 随机变量

随机变量是随着试验的结果的不同而变化的数,即它是样本点的一个函数。例如有一个方程,如下:

$$y=5x-2 \quad (2-32)$$

式(2-32)中有 x 和 y 两个变量。我们知道如果已知其中一个(例如 $y=8$),那么就能够求解出另一个($x=2$)。然而,随机变量却与此很不相同,随机变量允许我们对随机事件指派数值。例如,想要对抛硬币的结果建模。开始时设置一个变量 X ,如果正面朝上赋给 X 值1,否则为0。 X 是一个随机变量——“变量”部分描述了 X 可以取不同值的事实(这里是0、1),而所谓的“随机”是指在抛掷硬币发生之前我们不知道 X 取什么值。

我们不能用与标准变量函数相类似的方式来表达这个结果(例如, $y=5f-2$)。按照惯例,我们用大写字母来表示随机变量,用小写字母来表示这个随机变量可以取的值。随机变量有两种类型,并且这两种随机变量处理方式略有不同。离散随机变量是最容易理解的概念,它们用来表示随机事件,对于这些随机事件,我们可以系统地列出随机事件所有可能的结果。例如,离散随机变量可以用来描述投硬币(可能的结果为0、1),或者掷骰子(可能的结果为1~6)。这些可能结果的集合叫作样本空间。

有组织地按顺序写出的事件似乎应该适用于任何事情。实际并非如此。以奥运会男子100米短跑为例,假设获胜时间为9~10秒。

我们尝试系统地写下所有的可能:

9, 9.1, 9.2, ...

在某一点上,我们会意识到我们错过了一些数据,例如,所有 9~9.1 之间的可能数据,因此我们现在重新开始:

9, 9.01, 9.02, ..., 9.1, ...

但是 9~9.01 的值怎么办? 第三次尝试:

9, 9.001, 9.002, 9.003, ..., 9.01,

这个事件所有可能的结果不能够系统地写出来(每次写下两个值之后,两个值之间都有一些丢失的值)。对于这种事件,我们需要使用连续随机变量。

表 2-1 给出了我们想要使用随机变量建模的事件或者量,有离散的也有连续的。下面我们通过离散随机变量介绍几个重要概念,之后再将这些概念扩展到连续随机变量情形。

表 2-1 随机变量建模的事件

过程	离散或连续
抛硬币	离散
投骰子	离散
100 米比赛的结果	连续
计算机网络中的失效节点	离散
诉讼案件的结果	离散
人的身高	连续
卵石的质量	连续
足球比赛的得分	离散

2.7.2 概率和概率分布

设 Y 是表示抛硬币的随机变量。正面,则 $y=1$;反面,则 $y=0$ 。要对这个事件建模,需要能够量化每一个结果的可能性。对于离散随机变量,我们通过定义不同结果的概率来实现。考虑某个特定结果概率的直观方法是假设它表示该结果出现的次数与事件重复次数的比例。如果公平的抛(即两面朝上可能

性一致)硬币 1000 次,那么我们期望看到正面向上的概率大约占一半(剩下的反面朝上)。将正面朝上的概率 $P(Y=1)$ 定义为一半或者 0.5 看起来是合理的。如果硬币不是正面朝上,那么它将反面朝上(在我们的样本空间中只有这两种可能),因此反面朝上的概率是 1 减去正面朝上的比例。因此, $P(Y=0) = 1 - P(Y=1) = 0.5$

多次重复实验时,以一个特定结果发生次数的比例作为概率不是我们能想到的定义概率的唯一方法。特别是对于某些只能出现一次的事件,它并不总是最自然的类比。

从对比例的简单讨论中,能够得到概率的两条重要规则。

(1) 概率必须大于或等于 0 (比例不能为负数) 同时小于或等于 1。

(2) 所有可能单个结果的概率之和必须等于 1。

例如抛硬币:

$$P(Y=1) + P(Y=0) = 1 \quad (2-33)$$

掷骰子:

$$P(Y=1) + P(Y=2) + \dots + P(Y=6) = 1 \quad (2-34)$$

这些陈述等同于数学表达式:

$$0 \leq P(Y=y) \leq 1 \quad (2-35)$$

$$\sum_y P(Y=y) = 1 \quad (2-36)$$

依据惯例,小写字母 y 表示随机变量 Y 可以取的所有可能值。注意,我们经常需要书写对随机变量所有取值求和,为了保持符号简明,用 $\sum_y$ 表示对随机变量 Y 所有可能的取值求和。

$P(Y=y)$ 是一个标量,即随机变量 Y 取值为 y 的概率。这个符号有时候是不方便的,因此我们有时候使用下面的简写:

$$P(Y=y) = P(y) \quad (2-37)$$

所有可能取值的集合(所有的 y)和它们的概率($P(y)$)称为概率分布。它表明总概率(1)如何在所有可能的结果上分配。

通常,我们使用式(2-35)和式(2-36)基于某些基本假设来定义概率。例如在抛硬币例子中,我们假设正、反两个结果是等可能的: $P(y=1) = P(Y=0) =$

r 。将这个假设代入式(2-36),并且已知 r 属于 $0 \sim 1$,我们可以使用基本代数计算出 r 的值。

2.7.3 概率的加法

设 Y 是一个随机变量,用来对掷公平骰子的结果建模。如果假设骰子是公平的,即所有结果是等可能的,通过之前的学习我们知道计算每一个结果(1、2、3、4、5、6)的概率,但掷一次骰子且点数是4,再次投掷骰子,点数小于4的概率是多少?小于4的所有点数是1、2、3,这表明我们能够计算骰子出现1、2、3点的概率。如果骰子已经投掷了许多次,那么我们能够计算某一点数出现的比例。出现1、2、3点数的比例等于出现1点的比例加上出现2点的比例以及出现3点的比例。这引导我们得出概率的加法定律:

$$P(Y < 4) = P(Y = 1) + P(Y = 2) + P(Y = 3) \quad (2-38)$$

无论我们感兴趣的结果的顺序如何,都得到了完全一致的答案。例如,投掷出1点或者6点的概率应该是 $P(Y = 1) + P(Y = 6)$ 。这并不仅仅限制在特定的结果上。例如,掷骰子出现不是4点的概率可以这样计算:

$$\begin{aligned} P(Y \neq 4) &= P(Y < 4) + P(Y > 4) \\ &= P(Y = 1) + P(Y = 2) + P(Y = 3) + P(Y = 5) + P(Y = 6) \end{aligned} \quad (2-39)$$

值得一提的是在上诉例子中,事实上使用式(2-36)计算更简单:

$$P(Y \neq 4) + P(Y = 4) = 1 \Rightarrow P(Y \neq 4) = 1 - P(Y = 4)$$

2.7.4 条件概率

在获取事件概率时,有时一个事件常常会影响另一个事件的结果。例如,抛一个硬币然后我告诉你结果(你看不到硬币)。这里有两个事件:第一个,抛硬币;第二个,我告诉你抛硬币的结果。假设这两个事件用两个随机变量 X 和 Y 表示。如果正面朝上, X 为1;反面朝上, X 为0。如果我告诉你正面 Y 为1;否则,为0。除非我行为怪异,否则, Y 的结果将依赖于 X 的结果。条件概率表达已知 X 取特定值的时候 Y 也取特定值的概率,表达式为

$$P(Y = y | X = x) \quad (2-40)$$

读作当 X 取值为 x 时, Y 取值为 y 的概率。与非条件概率一样,我们使用如下的简写:

$$P(Y=y|X=x)=P(y|x) \quad (2-41)$$

在上诉例子中,如果假设我总是说真话,那么硬币是正面我告诉你硬币是正面的概率是 1(总是发生):

$$P(Y=1|X=1)=1 \quad (2-42)$$

对于反面也是一样:

$$P(Y=0|X=0)=1 \quad (2-43)$$

使用式(2-36)和上述的概率,我们可以推导出 $P(Y=0|X=1)$ 和 $P(Y=1|X=0)$:

$$P(Y=0|X=1)=1-P(Y=1|X=1)=0 \quad (2-44)$$

$$P(Y=1|X=0)=1-P(Y=1|X=1)=0 \quad (2-45)$$

如果我不诚实,那么事情将会变得更有趣。假设硬币反面朝上我总是说真话,但是如果硬币是正面我说真话(即正面)次数的比例是 0.8。这意味着,如果硬币是正面我说正面的概率是 0.8,说反面的概率是 0.2。在这个假设下,所有的条件概率如下:

$$P(Y=1|X=1)=0.8 \quad (2-46)$$

$$P(Y=0|X=1)=0.2 \quad (2-47)$$

$$P(Y=1|X=0)=0 \quad (2-48)$$

$$P(Y=0|X=0)=1 \quad (2-49)$$

与非条件概率一样,概率必须满足式(2-36), $\sum_y P(Y=y|X=x)=1$ 。检查刚计算的值:

$$\sum_y P(Y=y|X=1)=P(Y=1|X=1)+P(Y=0|X=1)=0.8+0.2=1 \quad (2-50)$$

$$\sum_y P(Y=y|X=0)=P(Y=1|X=0)+P(Y=0|X=0)=0+1=1 \quad (2-51)$$

根据条件概率并假设 $P(X=1)=P(X=0)=0.5$ (即硬币是公平的),我们可能会问:“硬币是正面,我说正面的概率是多少?”这与 $P(Y=1|X=1)$ 不同;这个条件概率假设 $X=1$ 已经发生,唯一不确定的是剩下的 Y 将是什么结果。然而我的问题关注这两个事件($X=1$ 和 $Y=1$)。如果都没有发生,那么它们同

时取得特定结果的概率是多少? 我们可能需要评估其他相关的量, 如 $P(Y=1)$ 和 $P(Y=0)$, 即我说硬币是正面或者反面的概率。为此, 需要多元概率和多项分布的知识, 这些知识将在下一小节进行讲解。

2.7.5 联合概率

已知 2 个(或更多)随机变量, 我们可能想知道它们每个取得某一特定值的概率。现继续以之前抛硬币为例开展这一方面的讨论。我们可能想要知道硬币是正面同时我说正面或者硬币是反面同时我说正面的概率。这些是联合概率, 定义为

$$P(Y=y, X=x) \quad (2-52)$$

式(2-52)也可表示成函数形式 $P(y, x)$ 。我们处理联合分布的依据主要在于这些随机变量是否相关。在例子中, Y (我说的结果)依赖于 X (抛硬币的结果)。即使我不那么诚实, 抛硬币的结果也决定我说什么。如果两个变量没有相关性(例如, 两个随机变量表示不同的抛硬币事件, 一个结果不可能影响另一个结果), 那么联合概率可以通过两个单独概率相乘来计算:

$$P(Y=y, X=x) = P(Y=y) \times P(X=x) \quad (2-53)$$

Y 取值 y 并且 X 取值 x 的概率等于 Y 取值 y 的概率乘以 X 取值 x 的概率。更一般地(为了方便), 我们使用函数形式 $p(y_1, \dots, y_J)$, 而不是 $P(Y_1 = y_1, \dots, Y_J = y_J)$, 对于 J 个随机变量, 有

$$P(y_1, y_2, \dots, y_J) = P(y_1) \times P(y_2) \times \dots \times P(y_J) = \prod_{j=1}^J P(y_j) \quad (2-54)$$

如果这些事件是相互依赖的, 那么我们就不能以这种方式分解联合概率。如果能建立条件概率分布, 我们就可以使用如下定义来分解联合概率:

$$P(Y=y, X=x) = P(Y=y|X=x) \times P(X=x) \quad (2-55)$$

或者

$$P(Y=y, X=x) = P(X=x|Y=y) \times P(Y=y) \quad (2-56)$$

所以, 硬币正面朝上并且我说正面的概率是

$$P(Y=1, X=1) = P(Y=1|X=1) \times P(X=1) = 0.8 \times 0.5 = 0.4 \quad (2-57)$$

换句话说, 如果重复多次实验, 那么硬币朝上并且我说朝上的比例是 0.4。

我偶尔撒谎的事实使得硬币是正面你也听到正面的概率从 0.5 (我诚实的情况下) 下降至 0.4。

X 和 Y 有 4 种可能的组合, 因此有 4 种可能的结果。式(2-36)表明: 如果将这 4 种情况的概率相加, 和应该是 1。

$$\sum_{x,y} P(X=x, Y=y) = 1 \quad (2-58)$$

在式(2-58)中, $\sum$ 对应着对 x, y 所有 4 种可能组合情况求和。我们可以使用式(2-55)计算所有情况来检测式(2-58)。我们已经知道 $P(X=1, Y=1) = 0.4$ 。其余的是:

$$P(Y=0, X=1) = P(Y=0|X=1)P(X=1) = 0.2 \times 0.5 = 0.1$$

$$P(Y=1, X=0) = P(Y=1|X=0)P(X=0) = 0 \times 0.5 = 0$$

$$P(Y=0, X=0) = P(Y=0|X=0)P(X=0) = 1 \times 0.5 = 0.5 \quad (2-59)$$

将这些概率加在一起, 即 $0.4 + 0.1 + 0 + 0.5 = 1$, 表明计算结果是正确的。在进一步学习之前, 我们需要明确这三个值的意义。第一个值(0.1)给出了硬币是正面我说是反面的概率。这比我诚实情况下对应的概率有所增加(我总是说真话时的概率是 0), 因为硬币是正面时我偶尔撒谎。第二个值(0)给出了硬币实际是反面而我说是正面的概率。这个值是 0, 因为硬币是反面的时候我不撒谎。第三个值给出了硬币是反面而我也说是反面的概率。它是 0.5, 因为硬币有一半的次数是反面, 而反面的时候我总是讲真话。

2.7.6 边缘化

如果记录我说正面或者反面次数的比例, 实际上是计算 $P(Y=1)$ 和 $P(Y=0)$ 。这些表达式没有包含 X 。 $P(Y=y)$ 可以通过从联合概率 $P(Y=y, X=x)$ 中边缘化 X 得到。通过对联合概率在 X 的所有可能取值上求和可得

$$P(Y=y) = \sum_x P(Y=y, X=x) \quad (2-60)$$

在我们硬币的例子中, X 可以取两个值(0,1)中的一个, 所以求和就变成:

$$P(Y=y) = P(Y=y, X=0) + P(Y=y, X=1) \quad (2-61)$$

一般而言, 对于 J 个随机变量的联合概率, 为了获得它们中一个的边缘分布 $P(Y_i=y_i)$, 可以通过如下的公式:

$$P(y_j=y_j) = P(y_j) = \sum_{y_1, \dots, y_{j-1}, y_{j+1}, \dots, y_J} \quad (2-62)$$

表达式表示对剩余 $J-1$ 个变量(缺少 y_i)的所有可能情况求和。例如,如果 $J=3$ 并且每个变量只能取值 0、1,那么为了计算 $P(Y_1=y_1)=p(y_1)$ 就需要对 y_2 和 y_3 的四种不同组合求和:

$$\begin{array}{cc} y_2 & y_3 \\ 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{array} \quad (2-63)$$

如果 $J=4$,那么这个数就增加到 8:

$$\begin{array}{ccc} y_2 & y_3 & y_4 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{array} \quad (2-64)$$

一般而言,对于二元变量,组合的数量是 2^{J-1} ,它随着 J 呈指数增长。如果随机变量有 2 个以上的结果,那么情况更坏(例如,对于骰子 6^{J-1})。在某些机器学习的概率领域,边缘化非常重要,也具有挑战性。继续回到硬币的例子, $P(Y=1)$ 是,则

$$P(Y=1)=\sum_x P(Y=1, X=x)=0+0.4=0.4 \quad (2-65)$$

并且 $P(Y=0)$ 是,则

$$P(Y=0)=\sum_x P(Y=0, X=x)=0.5+0.1=0.6 \quad (2-66)$$

我们也可以使用 $P(Y=1)$ 的值和式(2-36)计算 $P(Y=0)$ 的值。这些概率表明,我说正面和反面的次数占总次数的比例。这不同于抛硬币得到正面或者反面的次数占总次数的比例($P(X=1)=P(X=0)=0.5$)。这一矛盾是由于在结果告知过程中的不确定性引起的。

2.7.7 贝叶斯规则

在机器学习中,通常我们感兴趣的是在给定训练数据 D 时,确定假设空间 H 中的最佳假设。贝叶斯理论提供了一种直接计算这种可能性的方法。更精确地讲,贝叶斯法则提供了一种计算假设概率的方法,它基于假设的先验概率、给定假设下观察到不同数据的概率以及观察到的数据本身。贝叶斯推理的问题是条件概率推理问题,这一领域的探讨对揭示人们对概率信息的认知加工过程与规律、指导人们进行有效的学习和判断决策都具有十分重要的理论意义和实践意义。鉴于之后章节将对这部分知识做较深入的探讨,本小节仅对贝叶斯规则做粗略介绍。

在式(2-55)的左侧和式(2-56)的左侧是相同的,因此右侧也是相等的,即

$$P(Y=y|X=x)P(X=x)=P(X=x|Y=y)P(Y=y) \quad (2-67)$$

重新整理,可以得到 $Y=y$ 条件下 X 的概率 $P(X=x|Y=y)$,这依赖于 $X=x$ 条件下 Y 的概率。这就是著名的贝叶斯规则:

$$P(X=x|Y=y)=\frac{P(Y=y|X=x)P(X=x)}{P(Y=y)} \quad (2-68)$$

在抛硬币的例子中,现在已经知道结果, X 取得特定值的概率(或以 $Y=y$ 为条件, $X=x$ 的概率)。但如果想要预测硬币实际的正、反面情况,则需要进行以下计算。

代入数值可以计算出 $P(X=1|Y=1)$:

$$P(X=1|Y=1)=\frac{P(Y=1|X=1)P(X=1)}{P(Y=1)}=\frac{0.8 \times 0.5}{0.4}=1 \quad (2-69)$$

从上式也可以推导出

$$P(X=0|Y=1)=0 \quad (2-70)$$

类似地,可以计算

$$P(X=0|Y=0)=\frac{P(Y=0|X=0)P(X=0)}{P(Y=0)}=\frac{1 \times 0.5}{0.6}=0.83 \quad (2-71)$$

$$P(X=1|Y=0)=0.17 \quad (2-72)$$

式(2-69)和式(2-70)的值给出的是我说正面(即 $Y=1$)时真实的抛硬币的

概率,式(2-71)和式(2-72)的值是我说反面($Y=0$)时抛硬币的真实概率。 $P(X=1|Y=1)=1$ 说明,我说正面意味着正面是抛硬币的真实结果。 $P(X=0|Y=0)=0.83$ 表示,如果你听到的是反面,那么硬币出现反面(概率是0.83)比出现正面(概率是0.17)的可能性大很多。建模时这种方法逆转条件非常有用。

2.7.8 期望值

当处理随机变量时,使用一个或者多个值来代表一个分布的特征非常有用。例如,均值——我们期望随机变量采用平均值。期望值是随机试验在同样的机会下重复多次的结果计算出的等同“期望”的平均值,定义如下(离散随机变量):

$$E_{P(x)}\{f(X)\} = \sum_x f(x)P(x) \quad (2-73)$$

例如,如果我们对 X 的期望值(均值)感兴趣,那么 $f(X)=X$,并且表达式变为

$$E_{P(x)}\{X\} = \sum_x xP(x) \quad (2-74)$$

对于公平的骰子($P(x)=1/6$), X 的期望值是

$$E_{P(x)}\{X\} = \sum_x x \frac{1}{6} = \frac{1}{6} + \frac{2}{6} + \dots + \frac{6}{6} = \frac{21}{6} = 3.5 \quad (2-75)$$

从这个例子中我们注意到,期望值不需要是随机变量可能取值中的一个(如上述例子中,随机变量取值不可能是3.5点)。

其他函数的期望值可以以完全相同的方式计算。例如, $f(x)=X^2$ 的期望值

$$E_{P(x)}\{X^2\} = \sum_x x^2 \frac{1}{6} = \frac{1}{6} + \frac{4}{6} + \dots + \frac{36}{6} = \frac{91}{6} \quad (2-76)$$

随机变量的函数的期望值通常不是函数在 X 的期望值处的取值,明白这一点很重要。数学上, $E_{P(x)}\{f(X)\}$ 不一定等于 $f(E_{P(x)}\{X\})$ 。例如,我们刚才计算了 $E_{P(x)}\{X^2\}=91/6$,它不等于 $(E_{P(x)}\{X\})^2=(21/6)^2$ 。这两个值在一种情况下相等:当随机变量 X 的函数是一个常数乘以 X 时。在这种情况下,通过简单的代数运算就可以证明这两个值相等。

设 $f(X)=aX$,则

$$E_{P(x)}\{f(X)\} = \sum_x a x P(x) = f(E_{P(x)}\{X\}) \quad (2-77)$$

另一个重要的情况是,当函数仅仅是一个常数时。这时,因为概率分布对所有可能的结果求和必须为 1,所以期望值不再存在。例如 $f(X) = a$, 则

$$E_{P(x)}\{f(X)\} = \sum_x a P(x) = a \quad (2-78)$$

最后一种特殊情况是,函数和的期望值等于每个函数期望值的和(这种情况在应用中非常有用):

$$E_{P(x)}\{f(X)\} + g(X) = \sum_x (f(x) + g(x))P(x) = E_{P(x)}\{f(X)\} + E_{P(x)}\{g(X)\} \quad (2-79)$$

两种最常见的期望是均值(上面定义的 $E_{P(x)}\{X\}$)和方差。方差定义为实际值与期望值之差平方的期望值:

$$\text{var}\{X\} = E_{P(x)}\{(X - E_{P(x)}\{X\})^2\} \quad (2-80)$$

展开括号里的项,得出随机变量方差的表达式:

$$\text{var}\{X\} = E_{P(x)}\{X^2\} - 2E_{P(x)}\{X\}E_{P(x)}\{X\} + E_{P(x)}\{X\}^2 \quad (2-81)$$

从第二行到第三行,我们使用了 $E_{P(x)}\{E_{P(x)}\{f(X)\}\} = E_{P(x)}\{f(X)\}$ 。 $E_{P(x)}\{f(X)\}$ 的值是一个常数(通过求期望值消除了所有包含 X 的项),外层的期望是一个常数的期望值,我们前面已经说明,常数的期望值等于这个常数。将 $E_{P(x)}\{X\}^2$ 的项合并,我们得到

$$\text{var}\{X\} = E_{P(x)}\{X^2\} - E_{P(x)}\{X\}^2 \quad (2-82)$$

一般来说,方差大的随机变量的值比方差小的随机变量的值离均值更远,向量随机变量的期望值通常以完全相同的方式计算。对一个取向量值 x 的随机变量 X ,其期望值定义如下:

$$E_{P(x)}\{f(x)\} = \sum_x f(x)P(x) \quad (2-83)$$

其中,求和符号表示对向量 x 的所有可能值求和。因此,均值向量定义如下:

$$E_{P(x)}\{x\} = \sum_x xP(x) \quad (2-84)$$

处理向量的时候,方差的概念扩展为协方差矩阵,定义为

$$\text{cov}\{x\} = E_{P(x)}\{(x - E_{P(x)}\{x\})(x - E_{P(x)}\{x\})^T\} \quad (2-85)$$

如果 x 是一个 D 维的向量,那么 $\text{cov}\{x\}$ 是一个 $D \times D$ 维的矩阵。对角线

上的元素对应着 x 每个元素的方差, 对角线外的元素表示不同元素与 x 一起变化的程度, 即它们对于另外元素的依赖程度如何。

例如, 元素 x_d 和 x_e 对应一个大的正值, 如果 x_d 增加则 x_e 也增加。如果是一个大的负值, 则向相反方向移动(即 x_d 增加, 则 x_e 减小), 那么意味着他们是相关的。如果是 0 值或者接近 0 值, 意味着这两个元素没有关系(它们相互独立)。与方差一样, 协方差表达式可以表示为

$$\text{cov}\{x\} = E_{P(x)}\{xx^T - 2xE_{P(x)}\{x\}^T + E_{P(x)}\{x\}E_{P(x)}\{x\}^T\} \quad (2-86)$$

重新整理式(2-86)为

$$\text{cov}\{x\} = E_{P(x)}\{xx^T\} - E_{P(x)}\{x\}E_{P(x)}\{x\}^T \quad (2-87)$$

2.8 常见的离散分布

通过之前的学习, 我们能列出每一个随机变量的所有可能结果。但是随着可能结果数量的增加, 列出所有可能结果是不太可能的。在现实中, 我们常常需要处理许多著名的分布族。每一类分布适用于特定类型的事件, 通常, 这些分布使用参数来调节它们的特征。鉴于此, 本节将介绍一些在机器学习中常见的离散分布。

2.8.1 伯努利分布

在伯努利分布中, 有两个结果: 事件要么发生, 要么不发生。在介绍伯努利(Bernoulli)分布之前, 我们已经多次接触过它。如: 它在类似于抛硬币那样具有两个可能结果的事件的应用。对于随机变量 X , 可以取值为 0 或 1(二元随机变量), 将其取值为 1 的概率记为 q , 则伯努利分布可以表示如下:

$$P(X = x) = q^x (1-q)^{1-x} \quad (2-88)$$

伯努利分布是当 $N=1$ 时二项分布的特例。

2.8.2 二项分布

二项分布式扩展了伯努利分布, 用于定义 N 次试验中观察到的一定数目正面的概率。更一般地, 我们可以将它用于任何有两个结果(成功、失败)的事件上。如果有 N 个这类事件, 那么二项随机变量 Y 可以取从 0(没有一次成功) $\sim N(N$ 次都成功)的任意值。

观察到一定数目成功事件的概率由下式给出:

$$P(Y=y) = P(y) = \binom{N}{y} q^y (1-q)^{N-y} \quad (2-89)$$

表达式的第二部分看起来与伯努利分布表达式非常像。事实上,如果我们定义 N 个二元结果为 $x_1, x_2, \dots, x_N$, 那么二项分布表达式的第二部分是 N 个二项概率的乘积:

$$\prod_{n=1}^N q^{x_n} (1-q)^{1-x_n} = q^y (1-q)^{N-y} \quad (2-90)$$

其中, $y = \sum_n x_n$ 是成功的次数(成功对应 $x_n = 1$)。二项表达式的第一部分是必需的, 因为假设 $y = 3$, 那么有多个 $x_1, x_2, \dots, x_N$ 的可能组合与之对应。

$q^y (1-q)^{N-y}$ 只是表示了多种可能中的一个。计算所有可能结果的总和等于乘

以这些组合的数目, 已知组合数函数 $\binom{N}{y}$, 当 $N=50, q=0.7$ 时的分布函数如图2-9所示。

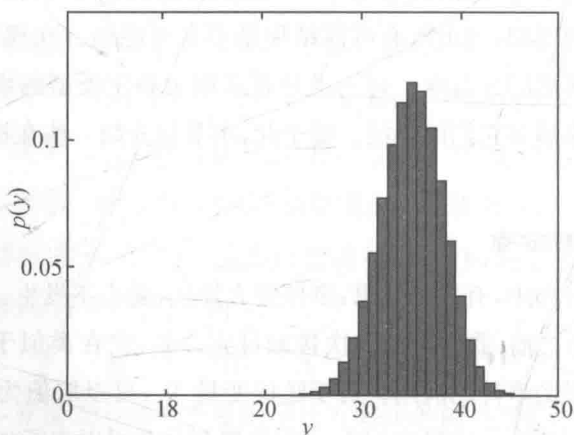


图 2-9 二项随机变量概率分布函数的示例

2.8.3 多项分布

前面两个分布实例均是标量随机变量的分布, 现在学习一种将概率分配给离散变量的向量的分布。基本思想与之前的基本相同, 分布函数对每一个可能的向量分配概率值, 这些概率的和必须为 1。例如: 假设你创建一个包含 N 个词的随机文本生成器, 并且你想要在这些随机文本上定义一个概率分布。使用词数的向量是表示文本的一种方式。

假设字典里有 J 个可能的词, 那么这个向量的长度为 J , 第 j 个元素保存

字典中第 j 个词在文本中出现的次数。设 Y 是一个表示文本的随机变量, 一个词数的向量 $y = [y_1, \dots, y_J]^T$ 是随机变量的一个实例, 多项分布定义 y 的分布如下:

$$P(Y=y) = P(y) = \frac{N!}{\prod_j y_j!} \prod_j q_j^{y_j} \quad (2-91)$$

其中, q_j 是多项式分布的参数, 表示第 j 个词的概率 ($\sum_j q_j = 1$)。

2.9 连续型随机变量——概率密度函数

在本章的开始部分我们已经发现, 不可能系统地写下连续随机变量所有可能的结果, 这导致我们难以对特定值分配概率。为了解决这个问题, 我们采用了一种方法, 即使用结果落入某个区间或者间隔的概率。例如, 已知一个连续随机变量 X 可以取负无穷大到正无穷大之间的任意值, 尝试计算出

$$P(x_1 \leq X \leq x_2) \quad (2-92)$$

当使用连续随机变量时, 我们需要概率分布的连续模拟(前面讲过, 对离散随机变量, 概率分布是一组结果(x)和每个结果的概率, 表示成 $x, p(x)$ 的函数)。这由概率密度函数(pdf)表示, 也记为 $p(x)$ 。为了计算 X 落入某一特定区间的概率, 我们计算 $p(x)$ 关于 x 在这个区间上的定积分

$$P(x_1 \leq X \leq x_2) = \int_{x_1}^{x_2} p(x) dx \quad (2-93)$$

如果随机变量只是可能在区间 $x_1 \leq X \leq x_2$ 上取值, 那么 X 落入这个区间的概率一定是 1。这引导我们利用式(2-36)对连续随机变量进行等价变形:

$$\int_{x_1}^{x_2} p(x) dx = 1_{x_1 \leq X \leq x_2} \quad (2-94)$$

式(2-35)也有一个连续随机变量的等价形式:

$$p(x) \geq 0 \quad (2-95)$$

它表明概率密度函数不能为负。值得注意的是, 概率密度函数没有上界, 因为概率密度函数不是概率, 所以对特定的 x , 其取值可以(常常是)比 1 大。

联合概率密度和条件连续概率密度: 与离散情况一样, 我们可以定义多个连续随机变量的联合概率密度函数。例如, $p(x, y)$ 是连续随机变量 X 和 Y 联合概率密度, $p(\omega)$ 是向量 ω 的概率密度函数, 向量 ω 的每个元素都是随机变

量, $p(\omega)$ 可以认为是 $p(\omega_0, \omega_1, \dots)$ 的联合概率密度函数。尽管我们不能计算 $P(X=x, Y=y)$, 但是我们可以计算

$$P(x_1 \leq X \leq x_2, y_1 \leq Y \leq y_2) = \int_{x=x_1}^{x_2} \int_{y=y_1}^{y_2} p(x, y) dx dy \quad (2-96)$$

同样可用于条件分布:

$$P(x_1 \leq X \leq x_2 | Y=y) = \int_{x=x_1}^{x_2} p(x | Y=y) dx \quad (2-97)$$

我们常常使用简写 $p(x|y)$ 来描述, 已知 $Y=y$ 时, X 的概率密度函数。

边缘化: 在联合概率中, 把最终结果中不需要的那些事件合并成其事件的全概率而消失(对离散随机变量用求和得全概率, 对连续随机变量用积分得全概率)称为边缘化(marginalization)。例如, 概率密度函数 $p(y)$ 可以由 $p(y, x)$ 计算出来:

$$p(y) = \int_{x=x_1}^{x_2} p(y, x) dx \quad (2-98)$$

其中, $x_1 \leq X \leq x_2$ 表示 X 的样本空间。

期望值: 连续随机变量的期望值是通过计算随机变量取值范围的积分来完成。

$$E_{p(x)}\{f(x)\} = \int f(x) p(x) dx \quad (2-99)$$

式(2-99)与 2.7.8 节中所有推导出的表达式在连续情形下的表示是相同的。

在某些实际情况中, 我们不知道 $p(x)$ 的确切形式, 或者它恰恰无法积分, 这导致我们可能不能计算其结果。对于这种情况, 我们通过对 $p(x)$ 进行采样, 然后通过下式近似得到:

$$E_{p(x)}\{f(x)\} \approx \frac{1}{S} \sum_{s=1}^S f(x_s) \quad (2-100)$$

其中, x 是从 $p(x)$ 采样得到的 S 个样本, 这是蒙特卡罗(Monte Carlo)近似的例子。此法由读者查阅相关文献自行学习。

2.10 常见的连续概率密度函数

与离散情形一样, 我们时常会碰到多类常见的连续概率密度函数。本节中, 我们介绍其中的 3 种。

2.10.1 均匀密度函数

最简单的连续密度函数是均匀密度函数。均匀密度函数 $p(y)=u(a,b)$, y 在 $a \sim b$ 之间时, 为常数, 其他为 0。

$$p(y)=\begin{cases} r(a \leq y \leq b) \\ 0(\text{其他}) \end{cases} \quad (2-101)$$

例如, 当 $a=3, b=8$ 时, 如图 2-10 所示。

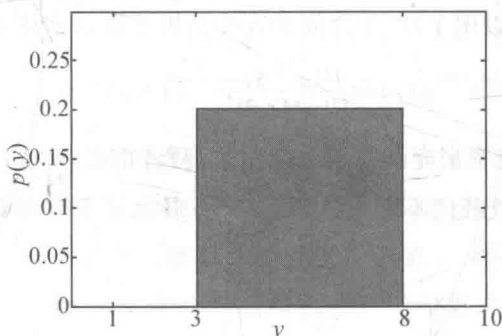


图 2-10 均匀概率密度函数的示例

根据概率密度函数在样本空间上的积分等于 1 的定义, 对于任意的 a, b , 我们能够计算出 r 值, 首先:

$$P(a \leq Y \leq d) = 1 = \int_{y=a}^b p(y) dy \quad (2-102)$$

然后根据式(2-101)和式(2-102)求出 r :

$$r = \frac{1}{b-a} \quad (2-103)$$

这是非常直观的(r 是全概率 1 除以区间长度), 因为随机变量必须在 $(b-a)$ 内。我们也可以很容易地定义多维均匀分布随机变量。例如, 如果 $y = [y_1, y_2]^T$

$$p(y)=\begin{cases} r(a \leq y_1 \leq b \text{ 且 } c \leq y_2 \leq d) \\ 0(\text{其他}) \end{cases} \quad (2-104)$$

并且可以以类似的方式计算 r 值:

$$P(a \leq y_1 \leq b, c \leq y_2 \leq d) = 1 = \int_{y_1=a}^b \int_{y_2=c}^d r dy_1 dy_2 = r(d-c)(b-a) \quad (2-105)$$

根据式(2-105)可得 r 的值:

$$r = \frac{1}{(d-c)(b-a)} \quad (2-106)$$

这也是直观的, r 是全概率 1 除以区间的面积, 随机变量必须落入面积 $(d-c)(b-a)$ 内。

2.10.2 β 密度函数

β 密度函数可以用于 $0 \sim 1$ 之间的连续随机变量, β 密度函数定义如下:

$$p(r) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} r^{\alpha-1} (1-r)^{\beta-1} \quad (2-107)$$

其中, α, β 是控制概率密度函数形状的参数, 两者都必须为正值。需指出 $\Gamma(z)$ 是伽马函数, 但此处我们不做讨论。图 2-11 展示了不同参数下的 β 概率密度函数。

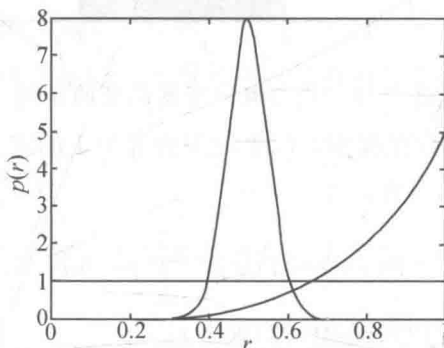


图 2-11 具有 3 对不同参数的 β 概率密度函数的示例

2.10.3 斯密度函数

高斯分布定义在实数域上(即 $-\infty \sim +\infty$), 随机变量 Y 的高斯概率密度函数定义为

$$P(y|\mu, \sigma^2) = \frac{1}{\sigma \sqrt{2\pi}} \exp\left\{-\frac{1}{2\sigma^2} (y-\mu)^2\right\} \quad (2-108)$$

上式的高斯概率密度函数由两个参数确定, 即: 均值(μ)和方差(σ^2)。图 2-12 展示了不同参数 μ, σ^2 的高斯密度函数。

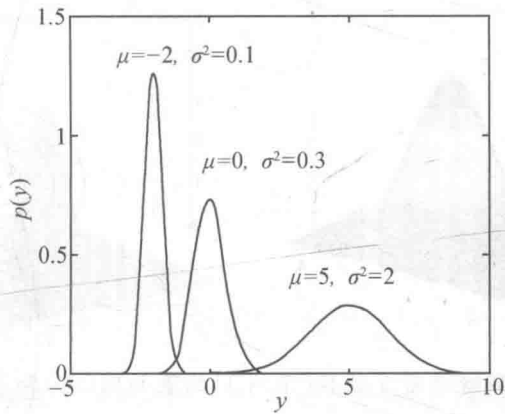


图 2-12 不同均值和方差的高斯概率密度函数

对于高斯概率密度函数,当 $y=\mu$ 时,概率密度函数取最大值,并且概率密度函数关于此点对称。概率密度函数的宽度由参数 σ^2 控制,值越大,密度越宽。如果我们使用图 2-12 最左边的高斯概率密度函数来生成随机变量的实例,那么我们只能期望得到 -2 附近小范围的值。对于最右侧的高斯概率密度函数,我们将获得 5 周围较大范围的值。高斯概率密度函数常用的速记法 $N(\mu, \sigma^2)$ 表示。因此,如果 Y 服从高斯概率密度函数,那么我们可以记作

$$p(y|\mu, \sigma^2) = N(\mu, \sigma^2) \quad (2-109)$$

式(2-109)读作随机变量 Y 的概率密度函数是正态均值 μ 和方差 σ^2 (高斯和正态通常是可交换的)。

2.10.4 多元高斯

高斯分布也可以用于定义连续向量的概率密度函数,如向量 $x=[x_1, \dots, x_D]^T$ 的多元高斯概率密度函数。其概率密度函数定义为

$$p(x) = \frac{1}{(2\pi)^{D/2} |\Sigma|^{1/2}} \exp \left\{ -\frac{1}{2} (x-\mu)^T \Sigma^{-1} (x-\mu) \right\} \quad (2-110)$$

其中, μ 是向量(大小与 x 相同), d 表示向量第 d 个元素对应的均值。方差变为一个 $D \times D$ 的协方差矩阵。本节用图形化的例子反应参数 μ 和 Σ 对概率密度函数的影响,如在示例 1 中,两个参数如式(2-111)所示,函数表面图和等概率率线图如图 2-13 所示。

$$\mu = [2, 1]^T, \Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (2-111)$$

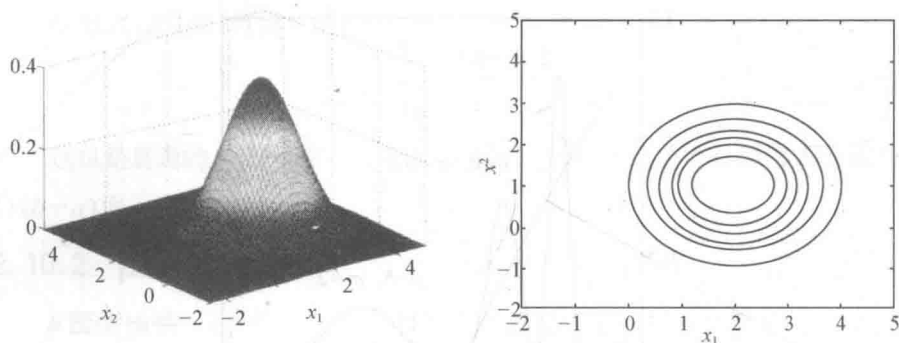


图 2-13 二元高斯概率密度函数(示例 1)的表面图(左)和等概率线图(右)

上述例子(示例 1)中的多元高斯只有两个变量(x_1 和 x_2)且 x_1 、 x_2 相互独立的特例。我们不难发现协方差矩阵 Σ 是单位矩阵 $\Sigma = I$ 。所以有

$$p(x) = \frac{1}{(2\pi)^{D/2} |I|^{1/2}} \exp \left\{ -\frac{1}{2} (x - \mu)^T I^{-1} (x - \mu) \right\} \quad (2-112)$$

由于 $I^{-1} = I$, 所以我们可获得单元高斯密度函数的乘积。从上述表达式开始(将 I^{-1} 替换为 I), 我们可以将指数内的矩阵积变成 D 个不同元素的和。由于和的指数等于指数的乘积, 所以最终可以将式(2-112)改写为

$$p(x) = \frac{1}{(2\pi)^{D/2} |I|^{1/2}} \prod_{d=1}^D \exp \left\{ -\frac{1}{2} (x_d - \mu_d)^2 \right\} \quad (2-113)$$

其中, $|I|$ 是矩阵 I 的行列式, 我们可以知道 $|I| = 1$ 。另一个常数项 $(2\pi)^{D/2}$, 可以记为 $\prod_{d=1}^D (2\pi)^{1/2}$, 所以表达式(2-113)还可以改写为

$$p(x) = \prod_{d=1}^D \frac{1}{(2\pi)^{1/2}} \exp \left\{ -\frac{1}{2} (x_d - \mu_d)^2 \right\} \quad (2-114)$$

在式(2-114)中, 乘积中的每一项是一个单元高斯分布(均值是 x_d , 方差是 1), 因此依据独立性定义, 向量 x 的每个元素相互独立。这个结论不仅仅适用于 $\Sigma = I$, 它还适用于协方差矩阵任意的对角线元素不为零的时候。这些对角线元素是每个单元高斯概率密度函数的方差。

在有些情况下, 我们不能把概率密度函数写成单元高斯概率密度函数的乘积, 这意味着向量 x 的每个元素不是相互独立的。例如示例 2, 其图像如图 2-14 所示, 2 个参数为:

$$\mu = [2, 1]^T, \Sigma = \begin{bmatrix} 1 & 0.8 \\ 0.8 & 1 \end{bmatrix} \quad (2-115)$$

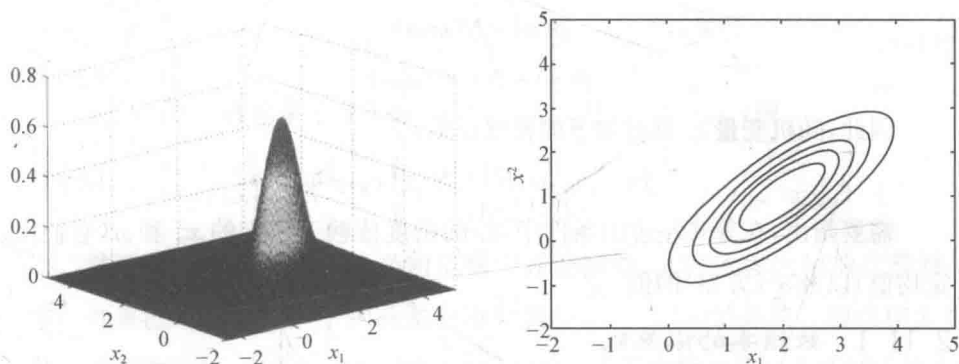


图 2-14 二元高斯概率密度函数(示例 2)的表面图(左)和等概率线图(右)

对于示例 2,我们在等概率线图中能看出它们之间的相关性,见图 2-14(右图)所示。如果 x_1, x_2 是独立的,那么 $p(x_2 | x_1)$ 不会随着 x_1 的不同而变化。当 $x_1=3$ 时, x_2 的值聚集在 2 周围。当 $x_1=1$ 时, x_2 的值聚集在 0 周围。很明显在两种情况下,期望 x_2 的值不同,可直观发现 x_1, x_2 是相关的。利用协方差矩阵中的值进行实验可以看出,对表面图和等概率线图的影响。

多元高斯的典型特征是它的条件概率密度函数 $p(x_2 | x_1)$ 是另一个很容易得到其均值和方差的高斯概率密度函数。尽管本节只是对此做了简要讲解,但这些知识对机器学习有着重要作用,是我们经常使用的方法。

2.11 似然估计

我们的模型如下所示:

$$t_n = f(x_n; \omega) + \epsilon_n \epsilon_n \sim N(0, \sigma^2) \quad (2-116)$$

如前面所述,该模型需要找到 ω 的最优值 $\hat{\omega}$,并且需要对可加性参数 σ^2 进行设置。在之前的学习中,我们找到了损失(该损失反应的是观测值 t 与模型预测值之间的差距)最少的 ω 值,同时给模型增加了一个随机变量,从而使模型的输出 t 自身也是一个随机变量。换句话中,对于一个特殊的 x_n ,其 t_n 的值不止一个。因此,我们不能用损失作为优化 ω 和 σ^2 的均值。

给高斯随机变量添加一个常量($\omega^T x_n$),实际就等同于具有相同常量转换获得均值的另一个高斯随机变量:

$$y = a + z$$

$$\begin{aligned} p(z) &= N(m, s) \\ p(y) &= N(m + a, s) \end{aligned} \quad (2-117)$$

因此,随机变量 t_n 具有如下的密度函数:

$$p(t_n | x_n, \omega, \sigma^2) = N(\omega^T x_n, \sigma^2) \quad (2-118)$$

需要指出,在等式左边的条件中, t_n 的密度依赖于特定的 x_n 和 ω (它们决定均值)以及 σ^2 (方差)的值。

2.11.1 数据集的似然值

一般来说,我们感兴趣的并非是单个数据点的似然值,而是整个数据集所有点的似然值。如果有 N 个数据点,我们感兴趣的就是它们的联合条件密度:

$$p(t_1, \dots, t_N | x_1, \dots, x_N, \omega, \sigma^2) \quad (2-119)$$

式(2-119)是数据集中所有点的联合密度(参见 2.7.5 节)。我们将利用 2.3 节定义的向量表示法和 X , 将其改写为紧缩形式为 $p(t | X, \omega, \sigma^2)$ 。这种估计观测值的密度可产生所有数据集的单个似然值,并且它可以通过改变 ω 和 σ^2 进行优化。

所有数据点的噪声是独立的($p(\epsilon_1, \dots, \epsilon_N) = \prod_n p(\epsilon_n)$),这一假设使我们能够将这种密度分解为更易操作的对象。即:分解为 N 个独立的部分,每一部分对应一个数据对象:

$$L = p(t | X, \omega, \sigma^2) = \prod_{n=1}^N p(t_n | x_n, \omega, \sigma^2) = \prod_{n=1}^N N(\omega^T x_n, \sigma^2) \quad (2-120)$$

在式(2-120), t_n 自身并非也是完全独立的。从平均变化来看,不难发现 t_n 随时间增加,表明了它们之间清晰的统计依赖性。另外,如果它们完全独立,也就根本不值得对数据进行建模。事实上,它们是条件独立的——对于给定的 ω 值(模型的决定性部分), t_n 是独立的,没有此条件则不独立。换句话说:假设我们搜集所有奥运会的年代(除了中间某一年,如 1960 年外)及获胜的时间。为了简便起见,我们使用 X, t 代表除了 1960 年外所有奥运会的年代和获胜时间。如果我们试图使用 X 和 t 来学习 t_{1960} , 那么我们感兴趣的是如下所示的条件分布:

$$p(t_{1960} | x_{1960}, X, t) \quad (2-121)$$

根据条件分布的定义,可以给出如下等式:

$$p(t_{1960} | x_{1960}, X, t) = \frac{p(t_{1960}, t | x_{1960}, X)}{p(t | X)} \quad (2-122)^*$$

假设参数 t 的元素是独立的,可得出 t_{1960} 仅依赖于 x_{1960} ,则

$$p(t_{1960} | x_{1960}, X, t) = \frac{p(t_{1960} | x_{1960}) \prod_n p(t_n | x_n)}{\prod_n p(t_n | X)} = p(t_{1960} | x_{1960}) \quad (2-123)$$

然而,在某种程度上 t_{1960} 必须依赖于其他数据,这种依赖性包括在参数 ω 中。如果已知 ω ,那么剩下的就是观测数据与 $\omega^T x_n$ 之间的差值。假设误差是独立的,因此以 ω 为条件,观测值也是独立的。为了使模型适用于任何应用,在下一节,我们将说明如何找到 ω 和 σ^2 的值来最大化似然值。

2.11.2 最大似然

在已知当前模型(即选定的 ω 和 σ^2)情况下,由于数据集是固定的,所以不断变化的模型将产生不同的似然值。模型的选择宗旨在于最大化似然值,换句话说,我们将选择那些能够使观测值最相似的模型参数值。出于统计的原因,我们使用似然函数的自然对数形式(我们将按照机器学习的惯例,使用 $\log(y)$ 来表示 y 的自然对数,而其他地方常常用 $\ln(y)$ 表示)。我这样做,因为估计的参数 $\hat{\omega}$ 和 $\hat{\sigma}^2$ 在最大化似然对数时,同样最大化了似然值。

取代高斯密度函数的表示形式(式(2-108))并分离各个变量,就得到了更加详细的表示形式:

$$\log L = -\frac{N}{2} \log_2 \pi - N \log \sigma - \frac{1}{2\sigma^2} \sum_{n=1}^N (t_n - f(x_n; \omega))^2 \quad (2-124)$$

用 $f(x_n; \omega) = \omega^T x_n$ 替换模型中的决定性部分,对数似然表达式就呈现如下形式:

$$\log L = -\frac{N}{2} \log_2 \pi - N \log \sigma - \frac{1}{2\sigma^2} \sum_{n=1}^N (t_n - \omega^T x_n)^2 \quad (2-125)$$

通过求导得到:

$$\frac{\partial \log L}{\partial \omega} = \frac{1}{\sigma^2} \sum_{n=1}^N x_n t_n - x_n x_n^T \omega = 0 \quad (2-126)$$

在式(2-126)中需要注意, $\omega^T x_n = x_n^T \omega$, $\frac{\partial \log L}{\partial \omega}$ 是向量,因此我们令其为0,即一个相同大小的为零的向量。

采用的简写矩阵/向量:

$$X = \begin{bmatrix} x_1^T \\ x_2^T \\ \dots \\ x_N^T \end{bmatrix} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ ? & \dots \\ 1 & x_N \end{bmatrix}, t = \begin{bmatrix} t_1 \\ t_2 \\ ? \\ t_N \end{bmatrix} \quad (2-127)$$

在表达式中, $\sum_{n=1}^N x_n t_n$ 可以写成 $X^T t$, 同理, $\sum_{n=1}^N x_n x_n^T \omega$ 可以写成 $X^T X \omega$, 我们用向量/矩阵形式写出其导数:

$$\frac{\partial \log L}{\partial \omega} = \frac{1}{\sigma^2} (X^T t - X^T X \omega) = 0 \quad (2-128)$$

解该表达式中的 ω , 可以得到最优值的表达式:

$$\omega = (X^T X)^{-1} X^T t \quad (2-129)$$

这就是 ω 的最大似然解, 为了获得 σ^2 的表达式 (假设 $\omega = \hat{\omega}$), 我们可以采用相同的步骤。采用偏导数和令其等于 0, 可以获得

$$\frac{\partial L}{\partial \sigma} = -\frac{N}{\sigma} + \frac{1}{\sigma^3} \sum_{n=1}^N (t_n - x_n^T \hat{\omega}) = 0 \quad (2-130)$$

重排给定的 $\hat{\sigma}^2, \sigma^2$ 的最大似然估计为

$$\hat{\sigma}^2 = \frac{1}{N} \sum_{n=1}^N (t_n - x_n^T \hat{\omega})^2 \quad (2-131)$$

式(2-131)是非常有意义的, 因为其变量就是简单的均方差误差。用矩阵表示时, 鉴于 $\sum_{n=1}^N (t_n - x_n^T \hat{\omega})^2$ 等于 $(t - X\hat{\omega})^T (t - X\hat{\omega})$, 于是:

$$\hat{\sigma}^2 = \frac{1}{N} \sum_{n=1}^N (t^T t - 2t^T X \hat{\omega} + \hat{\omega}^T X^T X \hat{\omega}) \quad (2-132)$$

此外, 还可以借助 $\hat{\omega} = (X^T X)^{-1} X^T t$ 进行进一步简化:

$$\hat{\sigma}^2 = \frac{1}{N} (t^T t - t^T X \hat{\omega}) \quad (2-133)$$

2.11.3 最大似然解的特点

在第 2.2 节中, 我们使用损失函数的 2 阶导数来确保我们已经找到最小值。现在用同样的方法, 用似然的 2 阶导数来确保我们已经找到最大的似然值。我们构建了 Hessian 矩阵, 其矩阵中的每个元素都是关于 ω 元素对的 2 阶

导数。为了确保我们已经找到了最大似然值,我们必须证实 Hessian 矩阵是负定的。

2 阶偏导数的 Hessian 矩阵可以通过对式(2-128)关于 ω^T 求导数,从而进行求解:

$$\frac{\partial^2 \log L}{\partial \omega \partial \omega^T} = -\frac{1}{\sigma^2} X^T X \quad (2-134)$$

如果用 $x_n = [1, x_n]^T$ 进行替换,那么该矩阵的对角线元素就等于(它们的不同之处在于乘以了一个常数)式(2-17)和(2-18)的 2 阶偏导数。这肯定是一个最大值,我们需要确定这个矩阵是否是负定的。我们可以通过如下来实现:

$$-\frac{1}{\sigma^2} z^T X^T X z < 0 \quad (2-135)$$

我们假设每个 x_n 是二维的,这样即可展开得到各项。我们将 X 进行定义为(与以前稍微不同):

$$X = \begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_N^T \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \\ \vdots & \vdots \\ x_{N1} & x_{N2} \end{bmatrix} \quad (2-136)$$

因此, $X^T X$ 为

$$X^T X = \begin{bmatrix} \sum_{i=1}^N x_{i1}^2 & \sum_{i=1}^N x_{i1} x_{i2} \\ \sum_{i=1}^N x_{i2} x_{i1} & \sum_{i=1}^N x_{i2}^2 \end{bmatrix} \quad (2-137)$$

式(2-137)与一个随机实向量 $z = [z_1, z_2]^T$ 进行前乘或后乘,结果为

$$z^T X^T X z = z_1^2 \sum_{i=1}^N x_{i1}^2 + 2z_1 z_2 \sum_{i=1}^N x_{i1} x_{i2} + z_2^2 \sum_{i=1}^N x_{i2}^2 \quad (2-138)$$

由于第一项和最后一项必须是正的,所以证明该表达式大于 0 就等同于证明它们的和大于中间项,即

$$z_1^2 \sum_{i=1}^N x_{i1}^2 + z_2^2 \sum_{i=1}^N x_{i2}^2 > 2z_1 z_2 \sum_{i=1}^N x_{i1} x_{i2} \quad (2-139)$$

定义 $y_{i1} = z_1 x_{i1}$, $y_{i2} = z_2 x_{i2}$, 并将其代入式(2-139),得

$$\sum_{i=1}^N (y_{i1}^2 + y_{i2}^2) > 2 \sum_{i=1}^N y_{i1} y_{i2} \quad (2-140)$$

对于任意的 i , 有

$$(y_{i1} - y_{i2})^2 > 0, \text{ 即 } y_{i1}^2 + y_{i2}^2 > 2y_{i1} y_{i2} \quad (2-141)$$

只有当 $y_{i1} = y_{i2}$, 故 $x_{i1} = x_{i2}$ 时该式才不成立——该情况在实际中不可能发生。如果对于任意 i , $y_{i1}^2 + y_{i2}^2 > 2y_{i1} y_{i2}$ 成立, 则任意数目对应该项的和也一定满足此不等式。因此 $z^T X^T X z$ 恒大于零, Hessian 矩阵是负定的, 且解就是最大似然值。

为了确保 $\hat{\sigma}^2$ 就是最大似然值, 我们对式(2-101)关于 σ 求导数:

$$\frac{\partial^2 \log L}{\partial \sigma^2} = \frac{N}{\sigma^2} - \frac{3}{\sigma^4} (t - X\hat{\omega})^T (t - X\hat{\omega}) \quad (2-142)$$

简化式(2-142), 结果为

$$\frac{\partial^2 \log L}{\partial \sigma^2} = -\frac{2N}{\sigma^2} \quad (2-143)$$

上式(2-143)恒小于零, 因此 $\hat{\sigma}^2$ 为最大似然值。

2.11.4 最大似然法适用于复杂模型

将 $\hat{\sigma}^2$ 的表达式(式(2-131))代入对数似然表达式(式(2-125)), 得到了最大值的对数似然值:

$$\log L = -\frac{N}{2} (1 + \log 2\pi) - \frac{N}{2} \log \hat{\sigma}^2 \quad (2-144)$$

该式告诉我们, L 的最大值随着 $\hat{\sigma}^2$ 的减小而增大, 其中, $\hat{\sigma}^2$ 是噪声的方差, 而噪声是模型的组成部分, 其目的在于捕获模型(即 $f(x; \omega)$)中决定性部分不能捕获的影响。减小 $\hat{\sigma}^2$ 的一种方法是调整 $f(x; \omega)$, 使其难以捕获数据中更多的变异性——使其更加灵活。

例如, 对于奥运会 100 米数据, 可以通过拟合阶数不断增大的多项式来增加似然值, 从而研究模型的灵活性(或复杂性)。图 2-15(a)表明 $\log(L)$ 随着多项式阶数的增大而增大, 并符合奥运会 100 米数据。如果我们打算用 $\log(L)$ 帮助我们选择使用哪个特定的模型, 那么它总是倾向于选择那些复杂度不断增大的模型。这看起来似乎是一个明智的决策——因为随着 $\hat{\sigma}^2$ 的减小, 模型能够捕获数据中更多的变异性。然而, 如果我们的任务是预测我们还没观察到的某

一年的获胜时间(比如 2016 年)。图 2-15(b)显示了 1 阶(虚线)和 8 阶(实线)多项式均适用于预测 2016 年的数据(用大的黑点表示)。更复杂的模型预测获胜的时间接近 11 秒(这可能是至今最慢的速度),而简单模型的预测更真实。从图中可以发现,似乎是简单模型捕获了数据中重要的关系(不断下降的趋势)而更复杂的模型却没有。这就是之前学习过的变化和过度拟合之间取舍的例子。

总之,简单模型比复杂模型具有更好的概括性,复杂模型过度拟合——我们给了其更大的空间以至于它试图将本质上是噪声的数据变得有意义。

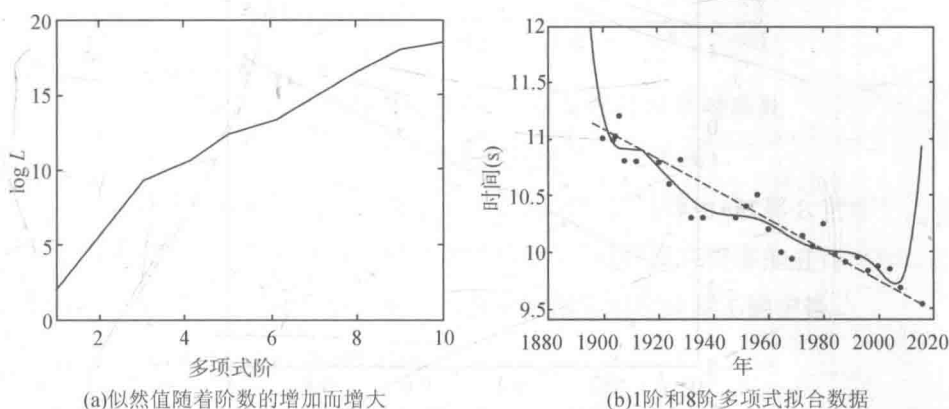


图 2-15 关于奥运会男子 100 米短跑数据模型复杂度的实例

2.12 噪声对参数估计的影响

2.12.1 参数估计的影响

在模型中噪声对参数估计有重要影响,如果噪声很大(σ^2 很高),模型可忍受 $\hat{\omega}$ 较大的改变,如果噪声很小,那么拟合的质量就将会快速恶化。所以在我们得到表达式前,有必要探究在生成综合数据过程中 $\hat{\omega}$ 的变化程度。本节通过生成大量具有相同真实 ω 和 σ^2 的数据来分析最大似然值估计 $\hat{\omega}$ 变化的方式。参考如下模型:

$$t_n = \omega_0 + \omega_1 x_n + \epsilon_n, \epsilon_n \sim N(0, \sigma^2) \quad (2-145)$$

假设真实参数值为 $\omega_0 = -2, \omega_1 = 3$, 且噪声方差 $\sigma^2 = 0.5^2$, 我们就能够对特定的一组特征值($x_1, x_2, \dots, x_N$)生成我们想要的任意多组数据($t_1, t_2, \dots$,

t_N), 并且能够计算每一组数据的 $\hat{\omega}$ 。图 2-16 显示了一个这种数据集的例子和务实方程, 其中特征值包含了 20 个 $(0, 1)$ 均匀分布的值, 即 $p(x) = u(0, 1)$ 。图 2-17 显示了生成的 10 000 个数据集以及每一种情况下的拟合值 $\hat{\omega}$ 。图 2-17 左图中的每个条形图形的高度表示生成某种特定范围内参数值的数据集的个数。图 2-17 的右图表示其相同含义的等概率线图。我们可以发现在真实值周围, ω_0 和均 ω_1 在较大范围内变化, 但很难从这些值得到该模型所呈现出来的变化程度。该例中 10 个数据集的 $\hat{\omega}$ 和真实函数如图 2-18 所示。

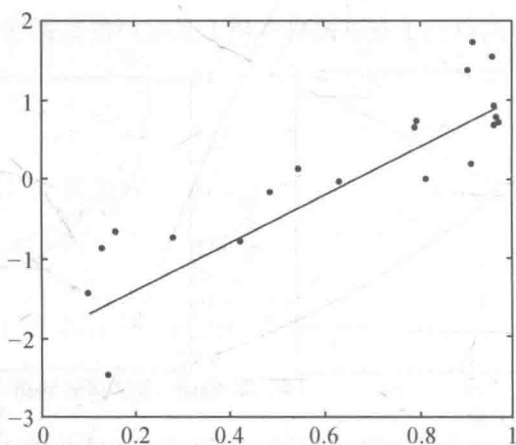


图 2-16 利用模型生成的数据和真实函数

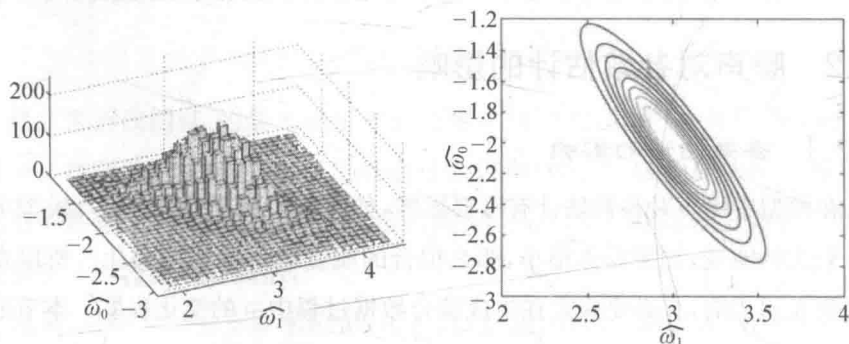


图 2-17 模型生成 10 000 个数据集的 $\hat{\omega}$ 变化情况

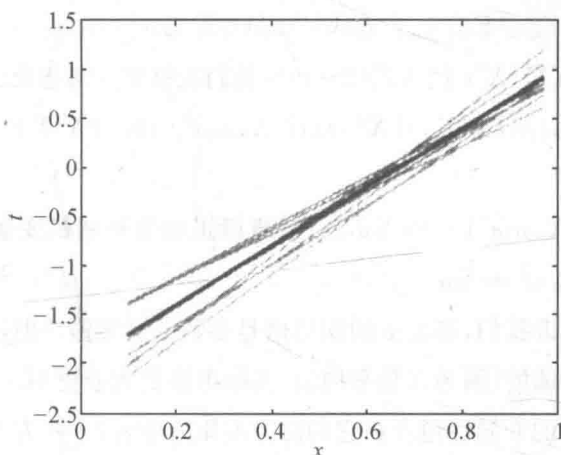


图 2-18 利用模型生成的 10 个数据集推导的函数

(注:真实函数为图中较粗较黑的直线)

如果假设真实数据集也是通过上述示例同样的过程生成,那么它对于定量估计结果的变异性很有帮助。只是通常情况下,我们没有很多能进行比较 ω 的数据集。接下来我们将介绍如何利用可用数据来确定这种不确定性。

2.12.2 参数估计的不确定性

从之前的学习,我们知道所获得的 $\hat{\omega}$ 值受数据中特定噪声严重影响。我们利用这一点,可有利于认识 $\hat{\omega}$ 的不确定性程度。为了更进一步探索,我们必须弄清楚 ω 和 $\hat{\omega}$ 的含义。利用之前已经提出与数据有关的模型,即

$$t_n = \omega^T x_n + \epsilon_n \quad (2-146)$$

其中, ω 表示参数的真实值, ϵ_n 是一个已经定义为正态分布的随机变量。这一假设意味着生成分布(或似然值) $p(t|X, \omega, \sigma^2)$ 是许多正态密度的乘积:

$$p(t|X, \omega, \sigma^2) = \prod_{n=1}^N p(t_n | x_n \omega, \sigma^2) = \prod_{n=1}^N N(\omega^T x_n, \sigma^2) \quad (2-147)$$

在 2.10.4 节中,我们已经介绍了如何将单元高斯密度的乘积写成一个具有对角协方差的多元高斯密度。由于用单个多元高斯密度比用多个单元高斯密度的积更简洁,所以多元高斯分布为

$$p(t|X, \omega, \sigma^2) = N(X\omega, \sigma^2 I) \quad (2-148)$$

目前, $\hat{\omega}$ 是真实参数值 ω 的估计。通过计算(见 2.7.8 节)可获得我们所期望的 $\hat{\omega}$, 其平均值为

$$E_{p(t|X,\omega,\sigma^2)}\{\hat{\omega}\} = \int \hat{\omega} p(t|X,\omega,\sigma^2) dt \quad (2-149)$$

将 $\hat{\omega} = (X^T X)^{-1} X^T t$ 代入式(2-149),我们能够估计其积分

$$E_{p(t|X,\omega,\sigma^2)}\{\hat{\omega}\} = (X^T X)^{-1} X^T \int t p(t|X,\omega,\sigma^2) dt = (X^T X)^{-1} X^T X \omega = \omega \quad (2-$$

150)

其中,由于 $p(t|X,\omega,\sigma^2) = N(X\omega, \sigma^2 I)$,所以正态分布随机变量的期望值等于其均值 $E_{p(t|X,\omega,\sigma^2)}\{t\} = X\omega$ 。

上述结果告诉我们,逼近 $\hat{\omega}$ 的期望值是参数的真实值。但这并不意味着我们的估计值是无偏的,因为这是平均值,实际可能更大或更小。

在 $\hat{\omega}$ 估计中的变异性包含在它的协方差矩阵中,这个协方差矩阵能够提供两方面的信息。即:(1)对角元素($\hat{\omega}$ 中单个元素的变异性)告诉我们单个参数中期望的变异性,即它们被数据定义的好坏程度。在前面的实验中,参数表现出较大的变异性,表明它们没有被数据很好地定义。(2)非对角元素告诉我们,参数如何协同变异,如果其值很高且为正,它就告诉我们增加一个值将导致其他值增大,从而可得到一个非常好的模型。大量负值则告诉我们相反的信息,即增大一个值导致其他值减小。趋近于零的值告诉我们这个参数与其他参数是独立的。例如,上文提到的例子(见图 2-17),似乎是增大 ω_1 ,导致 ω_0 下降,因此我们期望协方差矩阵中的非对角元素是负值。

2.13 预测值的变异性

我们之前做了一些关于未来奥运会 100 米获胜时间的预测。现在,我们认为这些预测并不合理,因为它们是以非常精确的形式呈现。在一定程度上,我们认为预测出一个取胜时间可能分布范围的值更加合理。如果我们非常确定我们的预测值,那么这个范围可能比较小。如果我们不太确定,范围可能就比较大。因此,正如我们得到参数估计值 $\hat{\omega}$ 的变异范围一样,得到预测值的变异范围或者不确定性是非常有意义的。假设我们观察了一组新的属性 x_{new} ,我们将要预测新的输出 t_{new} 以及其相应的变异度 σ_{new}^2 。

为了预测 t_{new} ,用将 x_{new} 乘以最优模型参数 $\hat{\omega}$:

$$t_{new} = \hat{\omega}^T x_{new} \quad (2-151)$$

其期望值

$$E_{p(t|X, \omega, \sigma^2)} \{t_{new}\} = E_{p(t|X, \omega, \sigma^2)} \{\hat{\omega}\}^T x_{new} = \omega^T x_{new} \quad (2-152)$$

利用通用的方差表达式, 可以获得

$$\sigma_{new}^2 = \{\text{var}\} \{t_{new}\} = E_{p(t|X, \omega, \sigma^2)} \{t_{new}^2\} - (E_{p(t|X, \omega, \sigma^2)} \{t_{new}\})^2 \quad (2-153)$$

为了评估该表达式, 我们首先需要将 $t_{new} = \hat{\omega}^T x_{new}$ 带入

$$\text{var}\{t_{new}\} = E_{p(t|X, \omega, \sigma^2)} \{x_{new}^T \hat{\omega} \hat{\omega}^T x_{new}\} - x_{new}^T \omega \omega^T x_{new} \quad (2-154)$$

代入类似 $\hat{\omega}$ 的表达式

$$\text{var}\{t_{new}\} = x_{new}^T (X^T X)^{-1} X^T E_{p(t|X, \omega, \sigma^2)} \{t t^T\} X (X^T X)^{-1} x_{new} - x_{new}^T \omega \omega^T x_{new} \quad (2-155)$$

使用 $\text{cov}\{t\}$ 的表达式, 可以计算并简化表达式

$$\begin{aligned} \text{var}\{t_{new}\} &= x_{new}^T (X^T X)^{-1} X^T (\sigma^2 I + X \omega \omega^T X^T) X (X^T X)^{-1} x_{new} - x_{new}^T \omega \omega^T x_{new} \\ &= \sigma^2 x_{new}^T (X^T X)^{-1} x_{new} \end{aligned} \quad (2-156)$$

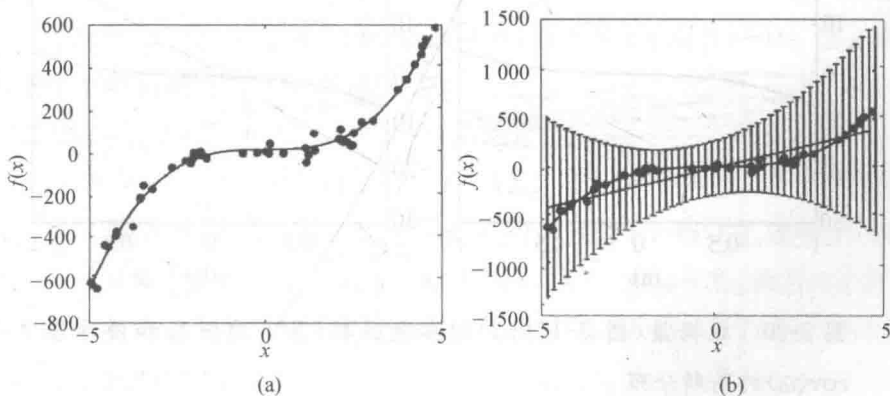
通过上述计算, 最后预测值和相应的方差为

$$\begin{aligned} t_{new} &= x_{new}^T (X^T X)^{-1} X^T t = x_{new}^T \hat{\omega} \\ \sigma_{new}^2 &= \sigma^2 x_{new}^T (X^T X)^{-1} x_{new} \end{aligned} \quad (2-157)$$

在式(2-157)中, σ^2 是数据噪声的真实方差。在此处, 我们可以用估计值 $\hat{\sigma}^2$ 替代它。

2.13.1 预测值变异性示例

图 2-19(a) 显示了方程 $f(x) = 5x^3 - x^2 + x$ 和取样点以及被均值为 0、方差为 1000 的高斯分布的噪声干扰的情况。在图 2-19(b)、(c)、(d) 中, 我们可以发现 $t_{new} \pm \sigma_{new}^2$ 分别为线性、立方和 6 阶模型。



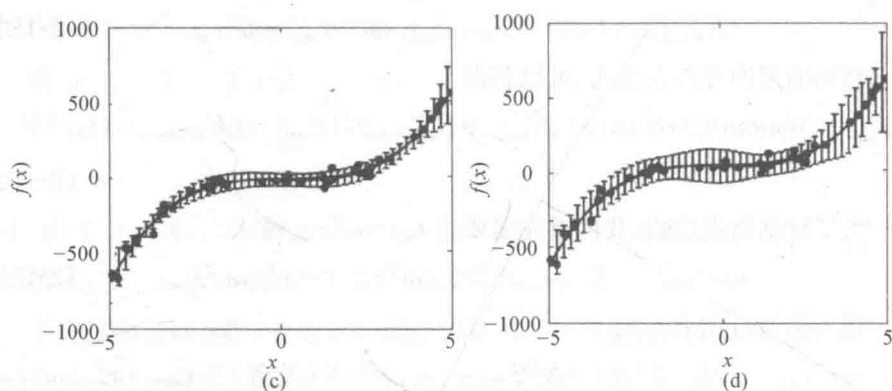


图 2-19 示例数据集和预测误差图

(注:a 为示例数据集;b、c、d 分别为线性、立方、6 阶模型的预测误差图)

尽管线性模型具有非常高的预测方差,但它不能非常好地对数据的真实趋势建模,并且数据的很多的变异性被假定为噪声。立方模型能够更好地对这些趋势建模(它是正确的阶数),并反映在了它更加可信的预测值中。6 阶模型过度复杂——它具有太大的随意度,因此能很好地拟合较大变化范围的参数值。 $\hat{\omega}$ 的这种不确定性可通过增加预测的变异性来实现,因为如果我们不能确定参数的估计值,那么我们也不能确定它的预测值。这一点可以通过计算 3 阶和 6 阶模型的协方差矩阵 $cov(\hat{\omega})$ 来证实。图 2-20 显示了 3 阶和 6 阶模型的 $\hat{\omega}$ 和 $cov(\hat{\omega})$ 高斯分布的函数图。在 6 阶模型中,清楚地显示了可能函数的不确定性随着参数不确定性的增长而增大的趋势。

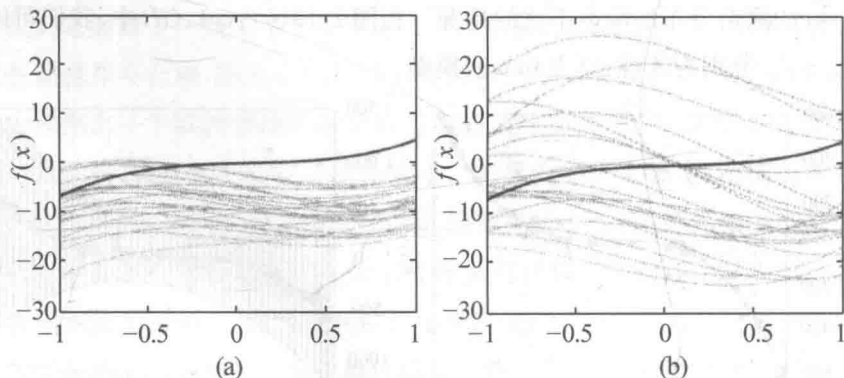


图 2-20 数据集(图 2-19(a))的参数函数,具有均值 $\hat{\omega}$ 和协方差 $cov(\hat{\omega})$ 的高斯分布

最后需要指出的是,对于所有的模型,预测值的方差都随着我们靠近数据的边缘而增大。

2.13.2 估计值的期望值

在2.12.2节中,我们计算了估计值 $\hat{\omega}$ 的期望值。该期望值用来生成 $p(t|X, \omega, \sigma^2) = N(X\omega, \sigma^2 I)$ 的密度,计算如下:

$$E_{p(t|X, \omega, \sigma^2)} \{\hat{\omega}\} = E_{p(t|X)} \{(X^T X)^{-1} X^T t\} = I\omega = \omega \quad (2-158)$$

在上述计算中,我们使用了 $\hat{\omega}$ 的表达式($\hat{\omega} = (X^T X)^{-1} X^T t$),并且高斯随机变量(t)的期望值等于高斯($X\omega$)的均值。因此,估计值 $\hat{\omega}$ 的期望值是真实值 ω 。这是 $\hat{\omega}$ 很重要的属性,它告诉我们 $\hat{\omega}$ 是一个无偏估计值,即它不会一直太高或太低。另一种考虑这点的方式是对于一组特征值 $x_1, x_2, \dots, x_N$,我们产生了一组响应值,并关注不同噪声对 $\hat{\omega}$ 有多大的影响。由于 $\hat{\omega}$ 是无偏的,所以平均来讲,它应该是正确的。因此,如果我们采用实验中所有 $\hat{\omega}$ 的平均值,那么将会非常接近真实值。事实上,我们采用 $\hat{\omega}_0 = -2.0007$ 和 $\hat{\omega}_1 = 3.0008$ 的平均值,它们都非常接近真实值: $\omega_0 = -2$ 和 $\omega_1 = 3$ 。

我们对噪声方差的估计 σ^2 可以采用同样的处理,式(2-133)中 $\hat{\sigma}^2$ 的表达式为

$$\sigma^2 = \frac{1}{N} (t^T t - t^T X\omega) \quad (2-159)$$

采用 $p(t|X, \omega, \sigma^2)$ 的期望值并进行一些操作,可以得到

$$E_{p(t|X, \omega, \sigma^2)} \{\hat{\sigma}^2\} = \frac{1}{N} E_{p(t|X, \omega, \sigma^2)} \{t^T t\} - \frac{1}{N} E_{p(t|X, \omega, \sigma^2)} \{t^T X (X^T X)^{-1} X^T t\} \quad (2-160)$$

前面我们已经看到了形如 tt^T 的表达式,但不是 $t^T t$ 或者 $t^T A t$ 。当 t 是高斯随机变量时,表达式 $t^T A t$ 的期望是

$$t \sim N(\mu, \Sigma) \\ E_{p(t)} \{t^T A t\} = \text{Tr}(A\Sigma) + \mu^T A \mu \quad (2-161)$$

其中, $\text{Tr}()$ 是迹函数,式中右边的第一项 $A = I_N$ (注意 $t^T t = t^T I_N t$,其中 I_N 是一个 $N \times N$ 的单位矩阵);第二项, $A = X(X^T X)^{-1} X^T$ 。在 $\mu = X\omega$ 和 $\Sigma = \sigma^2 I^N$ 在两种情况下,将必要的值代入式(2-160),并利用 $\text{Tr}(\sigma^2 A) = \sigma^2 \text{Tr}(A)$ 、 $\text{Tr}(I_N) = N$ 简化整理可得

$$E_{p(t|X, \omega, \sigma^2)} \{\hat{\sigma}^2\} = \sigma^2 \left(1 - \frac{1}{N} \text{Tr}(X(X^T X)^{-1} X^T) \right) \quad (2-162)$$

最后,我们需要利用 $\text{Tr}(AB) = \text{Tr}(BA)$ 的事实,将迹函数中的第一个 X 移动到最后:

$$E_{p(t|X, \omega, \sigma^2)} \{\hat{\sigma}^2\} = \sigma^2 \left(1 - \frac{1}{N} \text{Tr}((X^T X)^{-1} X^T X) \right) = \sigma^2 \left(1 - \frac{D}{N} \right) \quad (2-163)$$

在式(2-163)中, D 是特征的个数(X 中的列数)。

假设 $D < N$ (即我们测量的每一个数据点的特征数小于数据点的个数),那么平均方差的估计值平均比实际方差小:

$$E_{p(t|X, \omega, \sigma^2)} \{\hat{\sigma}^2\} < \sigma^2 \quad (2-164)$$

与 $\hat{\omega}$ 不同,该估计值是有偏的。我们可以返回到前面虚构的实验来观察这种偏差。所有数据集的 $\hat{\sigma}^2$ 的平均值为 0.226 4。真实值 $\hat{\sigma}^2 = 0.25$ 。我们可以发现,平均值确实太小了。对于这个例子, $D=2, N=20$, 因此我们理论上可以接受的期望值是

$$E_{p(t|X, \omega, \sigma^2)} \{\hat{\sigma}^2\} = 0.25 \left(1 - \frac{2}{20} \right) = 0.225 0$$

其接近于观察到的平均值。我们注意到降低这种偏置的一种方式使 D/N 变小。 D 一般是固定的,但我们可以增大 N 。在试验中我们可以看到将 N 从 20 增加到 10 000 的影响,即随着数据点的增加,理论曲线(虚线)及实验曲线(实线)非常接近且涵盖了真实值 $\hat{\sigma}^2 = 0.25$,如图 2-21 所示。

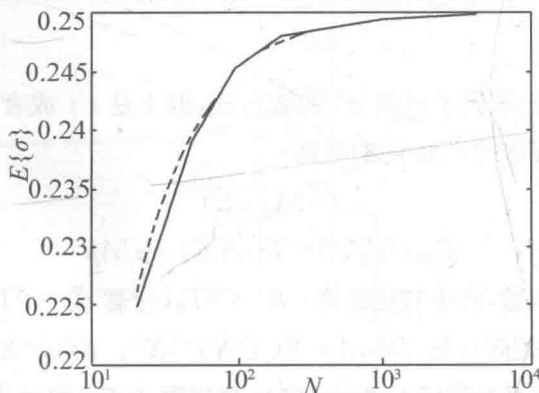


图 2-21 随着数据点的增加, $E_{p(t|X, \omega, \sigma^2)} \{\hat{\sigma}^2\}$ 的理论值与试验估计值的变化

上述试验可以提供一个关于 $\hat{\sigma}^2$ 偏差的直观解释。 σ^2 的最大似然值估计值的表达式是

$$\hat{\sigma}^2 = \frac{1}{N} (t^T t - t^T \hat{\omega}) \quad (2-165)$$

式(2-165)可改写为

$$\hat{\sigma}^2 = \frac{1}{N} (t_n - x^T \hat{\omega})^2 \quad (2-166)$$

上述表达式可告诉我们,模型越接近真实数据, $\hat{\sigma}^2$ 就越小。现在,思考一下 ω 的真实值与估计值 $\hat{\omega}$ 哪个更接近真实数据?根据定义,它们是一组接近真实数据的参数,因此可以最小化 $\hat{\sigma}^2$ 。如果式(2-165)中使用 ω 的真实值代替 $\hat{\omega}$,得到的 $\hat{\sigma}^2$ 的值将会大于或者等于使用 $\hat{\omega}$ 得到的值。因为我们将发现最小化噪声 ω 的平均值,所以将随着噪声程度低于真实值而中止。

2.14 评估假设

对假设的精度进行经验评估是机器学习中的基本问题。本小节重点介绍用统计方法估计假设精度,主要为解决以下三个问题:首先,已知一个假设在有限数据样本上观察到的精度,怎样估计它在其他实例上的精度?其次,如果一个假设在某些数据样本上好于另一个,那么一般情况下该假设是否更准确?再次,当数据有限时,怎样高效地利用这些数据,通过它们既能学习到假设,还能估计其精度?由于有限的数据样本可能不代表数据的一般分布,所以从这些数据上估计出的假设精度可能有误差。统计的方法,结合有关数据基准分布的假定,使我们可以用有限数据样本上的观察精度来逼近整个数据分布上的真实精度。

2.14.1 动机

多数情况下,对学习到的假设进行尽可能准确的性能评估十分重要。原因之一很简单,是为了知道是否可以使用该假设。例如,从一个长度有限的数据库中学习以了解不同医疗手段的效果,就有必要尽可能准确地知道学习结果的正确性。另一原因在于,对假设的评估是许多学习方法的重要组成部分。例如在决策树学习中,为避免过度拟合问题必须进行后修剪,这时我们必须评估每

一步修剪对树的精度产生的影响。因此,有必要了解已修剪和未修剪树的精度估计中固有的误差。

当数据十分充足时,假设精度的估计相对容易。然而当给定的数据集非常有限时,要学习一个概念并估计其将来的精度,存在两个很关键的困难。

(1)估计的偏差(Bias in the estimate):学习到的概念在训练样例上的观察精度通常不能很好地用于估计在将来样例上的精度。因为假设是从这些样例中得出的,因此对将来样例的精度估计通常偏于乐观。尤其在学习器采用了很大的假设空间并过度拟合训练样例时,这一情况就更可能出现。要对将来的精度进行无偏估计,典型的方法是选择与训练样例和假设无关的测试样例,并在这个样例集合上测试假设。

(2)估计的方差(Variance in the estimate):即使假设精度在独立的无偏测试样例上测量,得到的精度仍可能与真实精度不同,这取决于特定测试样例集合的组成。测试样例越少,产生的方差越大。

2.14.2 估计假设精度

在评估一个假设时,我们一般对估计这个假设对未来实例的类别的精度更感兴趣。同时也需要知道这一精度估计中的可能的误差(即与此估计相联系的误差门限)。本节使用的学习问题的框架如下:有一所有可能实例的空间 X (如所有人的集合),其上定义了多个目标函数(如计划本年购买滑雪板的人)。我们假定 X 中不同实例具有不同的出现频率,对此,一种合适的建模方式是,假定存在一未知的概率分布 D ,它定义了 X 中每一实例出现的概率(如 19 岁的人的概率比 109 岁的人的概率高)。注意 D 并没有说明 x 是一正例还是一反例,只确定了其出现概率。学习任务是在假设空间 H 上学习一个目标概念(即目标函数) f 。目标函数 f 的训练样例由施教者提供给学习器:每一个实例按照分布 D 被独立地抽取,然后它连同其正确的目标值 $f(x)$ 被提供给学习器。

为说明这一点,考虑目标函数“计划本年购买滑雪板者”,可以调查去滑雪板商店的顾客,通过此调查来收集训练样例。在这里实例空间 X 为所有人组成的集合,每个实例可由人的各种属性描述,如年龄、职业、每年滑雪次数等。分布情况 D 指定了在滑雪板商店中遇到每个人的概率。目标函数 $f: X \rightarrow$

$\{0,1\}$ 将每个人进行分类,判断它是否会在本年内购买滑雪板。

在这个一般的框架中,我们感兴趣的是以下两个问题。

(1)给定假设 h 和包含若干按 D 分布随机抽取的样例的数据集,如何针对将来按同样分布抽取的实例,得到对 h 的精度的最好估计。

(2)这一精度估计的可能的误差是多少?

1. 样本错误率和真实错误率

为解决上述的两个问题,需要确切地区分出两种精度(或两种错误率)。其一,是可用数据样本上该假设的错误率;其二,是在分布为 D 的整个实例集合上该假设的错误率。它们分别被称为样本错误率和真实错误率。对于从 X 中抽取的样本 S ,某假设关于 S 的样本错误率(sample error)是该假设错误分类的实例在 S 中所占的比例。

定义:假设 h 关于目标函数 f 和数据样本 S 的样本错误率(标记为 $\text{error}_s(h)$)为

$$\text{error}_s(h) = \frac{1}{n} \sum_{x \in S} \delta(f(x), h(x)) \quad (2-167)$$

其中, n 为 S 中样例的数量,而 $\delta(f(x), h(x))$ 在 $f(x) \neq h(x)$ 时为 1, 否则为 0。真实错误率(true error)是对于按 D 分布随机抽取的实例,该假设对它错误分类的概率。

定义:假设 h 关于目标函数 f 和分布 D 的真实错误率(由 $\text{error}_D(h)$ 表示),为 h 按 D 分布随机抽取实例被误分类的概率:

$$\text{error}_D(h) = Pr_{x \in D}[f(x) \neq h(x)] \quad (2-168)$$

这里,记号 $Pr_{x \in D}$ 表示概率在实例分布 D 上计算。

我们通常想知道的是假设的真实错误率 $\text{error}_D(h)$, 因为这是在分类未来样例时可以预料到的错误。然而我们所能测量的只是样本错误率 $\text{error}_s(h)$, 它所要求的数据样本 S 是我们所拥有的。本节所要考虑的主要问题就是“ $\text{error}_s(h)$ 在何种程度上提供了对 $\text{error}_D(h)$ 的估计?”。

2. 离散值假设的量信区间

为解决“ $\text{error}_s(h)$ 在何种程度上提供了对 $\text{error}_D(h)$ 的估计”的问题,先考虑 h 为离散值假设的情况。具体地说,比如我们要基于某离散值假设 h 在样本

S 上观察到的样本错误率估计它的真实错误率,其中:

(1) 样本 S 包含 n 个样例,它们的抽取按照概率分布 D ,抽取过程是相互独立的,并且不依赖于 h ;

(2) $n \geq 30$;

(3) 假设 h 在这 n 个样例上犯了 r 个错误(例如, $\text{errors}_S(h) = r/n$)。

已知这些条件,统计理论可给出以下结论:

(1) 没有其他信息的话, $\text{error}_D(h)$ 最可能的值为 $\text{errors}_S(h)$;

(2) 有大约 95% 的可能性,真实错误率 $\text{errors}_D(h)$ 处于下面的区间内:

$$\text{errors}_S(h) \pm 1.96 \sqrt{\frac{\text{errors}_S(h)(1 - \text{errors}_S(h))}{n}}$$

假如数据样本 S 包含 $n = 40$ 个样例,并且假设 h 在这些数据上产生了 $r = 12$ 个错误。这样,样本错误率为 $\text{errors}_S(h) = 12/40 = 0.30$ 。如果没有更多的信息,对真实错误率 $\text{error}_D(h)$ 的最好的估计即为样本错误率 0.30。然而我们不能期望这是对真实错误率的完美估计。如果另外搜集 40 个随机抽取的样例 S' ,样本错误率 $\text{errors}_{S'}(h)$ 将与原来的 $\text{errors}_S(h)$ 存在一些差别。这种差别是由 S' 和 S 组成上的随机差异所产生的。实际上,如果不断重复这一实验,每次抽取一个包含 40 个样例的样本 S_i ,将会发现约 95% 的实验中计算所得的区间包含真实错误率。因此,我们将此区间称为 $\text{error}_D(h)$ 的 95% 置信区间估计。在本例中, $r = 12$ 和 $n = 40$,根据上式,95% 置信区间为 $0.3 \pm (1.96 \times 0.07) = 0.3 \pm 0.14$ 。

上面的 95% 置信区间表达式可推广到一般情形以计算任意置信度。常数 1.96 是由 95% 这一置信度确定的。定义 z_N 为计算 $N\%$ 置信区间时的常数。计算 $\text{error}_D(h)$ 的 $N\%$ 置信区间的一般表达式为:

$$\text{errors}_S(h) \pm z_N \sqrt{\frac{\text{errors}_S(h)(1 - \text{errors}_S(h))}{n}} \quad (2-169)$$

其中, z_N 的值依赖于所需的置信度,参见表 2-2 中的取值。

表 2-2 双侧的 $N\%$ 置信期间的 z_N 值

置信度 $N\%$	50%	68%	80%	90%	95%	98%	99%
常量 z_N	0.67	1.00	1.28	1.64	1.96	2.33	2.58

因此,正如 $\text{error}_D(h)$ 的 95% 置信区间为 $0.30 \pm (1.96 \times 0.07)$ (其中 $r = 12, n = 40$), 可以求得同样情况下 68% 置信区间为 $0.30 \pm (1.0 \times 0.07)$ 。从直觉上我们也可以看出 68% 置信区间要小于 95% 置信区间, 因为我们减小了要求 $\text{error}_D(h)$ 落入此区间的概率。

公式(2-169)描述了为了在 $\text{error}_S(h)$ 基础上估计 $\text{error}_D(h)$, 如何计算置信区间(即误差门限)。这一表达式只能应用于离散值假设。它假定样本 S 抽取的分布与将来的数据抽取的分布相同, 并且假定数据不依赖于所测试的假设。还有, 该表达式只提供了近似的置信区间, 不过这一近似在至少包含 30 个样例并且 $\text{error}_S(h)$ 不太靠近 0 或 1 时很接近真实情况。判断这种近似是否接近真实, 更精确的规则为:

$$n \text{error}_S(h)(1 - \text{error}_S(h)) \geq 5 \quad (2-170)$$

上面我们概述了计算离散值假设的置信区间的过程, 下面将给出这一过程的统计学基础。

2.14.3 采样理论基础

本小节将介绍统计学和采样理论的几个基本概念, 包括概率分布、期望值、方差、二项分布和正态分布、双侧和单侧区间。在这些概念中, 有一些概念我们之前已经有了初步了解。但由于这些概念的正确理解对假设评估和算法评估有重要意见, 或者说, 它们提供了一种重要的概念框架, 以便于理解相关的机器学习问题(如过度拟合问题)以及理解在成功的泛化和训练样例数目之间的关系。所以, 本小结对这些概念做了进一步介绍, 已经熟悉这些概念的读者可以跳过本节。

1. 错误率估计和二项比例估计

样本错误率和真实错误率之间的差异与数据样本大小的依赖关系如何? 这一问题在统计学中已被透彻研究。在这里, 我们感兴趣的观察量为 h 是否误分类样例。

解决该问题首先要注意到, 测量样本错误率相当于在作一个有随机输出的实验。我们先从分布 D 中随机抽取出 n 个独立的实例, 形成样本 S , 然后测量样本错误率 $\text{error}_S(h)$, 如前一节所述, 如果将实验重复多次, 每次抽取大小为 n

的不同的样本 S_i , 将可以得到不同的 $\text{errors}_{S_i}(h)$ 的值, 它取决于不同 S_i 的组成中的随机差异。这种情况下, 第 i 个这样的实验的输出 $\text{errors}_{S_i}(h)$ ”。被称为一随机变量(random variable)。一般情况下, 可以将随机变量看成一个有随机输出的实验。随机变量值即为随机实验的观察输出。

设想要运行 k 个这样的随机实验, 测量随机变量 $\text{errors}_{S_1}(h), \text{errors}_{S_2}(h), \dots, \text{errors}_{S_k}(h)$ 。然后我们以图表的形式显示出观察到的每个错误率值的频率。当 k 不断增长, 该图表将呈现如图 2-22 所显示的分布。该表描述的概率分布称为二项分布(binomial distribution)。

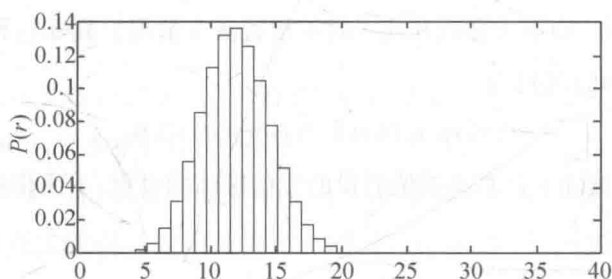


图 2-22 $n=40, p=0.3$ 时的二项式分布

一个二项分布给出了当单个硬币投掷出现正面的概率为 p 时, 在 n 个独立硬币投掷的样本中观察到 r 次正面的概率。它由以下的概率函数定义:

$$P(r) = \frac{n!}{r!(n-r)!} p^r (1-p)^{n-r} \quad (2-171)$$

如果随机变量 X 遵循二项分布, 则:

- (1) X 取值为 r 的概率 $Pr(X=r)$ 由 $P(r)$ 给出;
- (2) X 的期望值或均值 $E[X] = np$;
- (3) X 的方差 $Var(X) = np(1-p)$;
- (4) X 的标准方差 $\sigma_X = \sqrt{np(1-p)}$ 。

对于足够大的 n 值, 二项分布很接近于有同样均值和方差的正态分布。多数统计学家建议只在 $np(1-p) \geq 5$ 时使用正态分布来近似二项分布。

2. 二项分布

为较好地理解二项分布, 考虑以下的问题。有一磨损并弯曲了的硬币, 要估计在抛硬币时出现正面的概率。令此未知概率为 p , 投掷该硬币 n 次并计算

出现正面的次数 r 。对于 p 的一合理的估计为 r/n 。注意,如果重新进行一次该实验,生成一个新的 n 次抛硬币的集合,其出现正面次数 r 将与第一次实验有稍许不同,从而得到对 p 的另一个估计。二项分布描述的是对任一可能的 r 值(从 0 到 n),这个正面概率为 p 的硬币抛掷 n 次恰好出现 r 次正面的概率。

有趣的是,从抛掷硬币的随机样本中估计 p 与在实例的随机样本上测试 h 以估计 $\text{error}_D(h)$ 是相同的问题。一次硬币抛掷对应于从 D 中抽取一个实例并测试它是否被 h 误分类。一次随机抛掷出现正面的概率 p 对应于随机抽取的实例被误分类的概率(即 p 对应 $\text{error}_D(h)$)。 n 次抛掷的样本观察到 r 次正面,对应 n 个抽取的实例被误分类的数目。因此, r/n 对应 $\text{error}_S(h)$ 。估计 p 的问题等效于估计 $\text{error}_D(h)$ 。二项分布给出了一个一般形式的概率分布,无论用于表示 n 次硬币出现正面的次数还是在 n 个样例中假设出错的次数。二项分布的具体形式依赖于样本大小 n 以及概率 p 或 $\text{error}_D(h)$ 。

一般来说,应用二项分布的条件包括如下。

(1) 有一基本实验(如投掷硬币),其输出可被描述为一随机变量 Y 。随机变量 Y 有两种取值(如 $Y=1$ 为正面, $Y=0$ 反面)。

(2) 在实验的任一次尝试中 $Y=1$ 的概率为常数 p 。它与其他实验尝试无关。因此 $Y=0$ 概率为 $1-p$ 。一般 p 为预先未知的,面临的问题就在于如何估计它。

(3) 基本实验的 n 次独立尝试按序列执行,生成一个独立同分布的随机变量序列 $Y_1, Y_2, \dots, Y_n$, 令 R 代表 n 次试验中出现 $Y_i=1$ 的次数:

$$R = \sum_{i=1}^n Y_i \quad (2-172)$$

(4) 随机变量 R 取特定值 r 的概率(如观察到 r 次正面的概率)由二项分布给出:

$$\Pr(R=r) = \frac{n!}{r!(n-r)!} p^r (1-p)^{n-r} \quad (2-173)$$

二项分布刻画了 n 次硬币投掷出现 r 次正面的概率,也刻画了包含 n 个随机样例的数据样本出现 r 次误分类错误的概率。

3. 均值和方差

随机变量的两个最常用到的属性为其期望值(也称为均值)和方差。期望

值是重复采样随机变量得到的值的平均。更精确的定义如下。

定义:考虑随机变量 Y 可能的取值为 $y_1, \dots, y_n$, 期望值(expected value) $E[Y]$ 为

$$E[Y] = \sum_{i=1}^n Pr(Y=y_i) \quad (2-174)$$

例如,如果 Y 取值 1 的概率为 0.7, 取值 2 的概率 0.3, 那么期望值为 a 。如果随机变量 Y 服从二项分布, 那么可得

$$E[Y] = np \quad (2-175)$$

其中, n 和 p 为公式(2-173)中定义的二项分布的参数。

另一重要属性方差描述的是概率分布的宽度或散度, 即它描述了随机变量与其均值之间的差有多大。

定义:随机变量 Y 的方差(variance) $Var[Y]$ 为

$$Var[Y] = E[(Y - E[Y])^2] \quad (2-176)$$

方差描述的是从 Y 的一个观察去估计其均值 $E[Y]$ 的误差平方的期望。方差的平方根被称为 Y 的标准差, 记为 σ_Y 。

定义:随机变量 Y 的标准差(standard deviation) σ_Y 为

$$\sigma_Y = \sqrt{E[(Y - E[Y])^2]} \quad (2-177)$$

若随机变量 Y 服从二项分布, 则方差和标准差分别为

$$Var[Y] = np(1-p) \quad (2-178)$$

$$\sigma_Y = \sqrt{np(1-p)} \quad (2-179)$$

4. 估计量、偏差和方差

我们已得出随机变量 $error_S(h)$ 服从二项分布, 现在回到前面的问题: $error_S(h)$ 和真实错误率 $error_D(h)$ 之间可能的差异是多少?

用式(2-173)中定义二项分布的术语来描述 $error_S(h)$ 和 $error_D(h)$, 可得

$$error_S(h) = \frac{r}{n} \quad (2-180)$$

$$error_D(h) = p \quad (2-181)$$

其中, n 为样本 S 中实例数, r 是 S 中被误分类的实例数, p 为从 D 中抽取一实例被误分类的概率。统计学中将 $error_S(h)$ 称为真实错误率 $error_D(h)$ 的一个

估计量(estimator)。通常,估计量是用来估计某基本总体的某一参数的随机变量。对于估计量,显然最关心的是它平均来说是否能产生正确估计。下面我们定义估计偏差(estimation bias)作为估计量的期望值同真实参数值之间的差异。

定义:针对任意参数 p 的估计量 y 的估计偏差为

$$E[Y] - p \quad (2-182)$$

如果估计偏差为 0,我们称 Y 为 p 的无偏估计量(unbiased estimator)。注意,在此情况下由多次重复实验生成的 Y 的多个随机值的平均(即 $E[Y]$)将收敛于 p 。 $\text{error}_S(h)$ 是否为 $\text{error}_D(h)$ 的一个无偏估计量? 确实如此,因为对于二项分布, r 的期望值为 np 。因为 n 为一常数,那么 r/n 的期望值为 p 。

对估计偏差还需要作两点说明。首先,在本章开始我们提到,在训练样例上测试假设得到的对假设错误串的估计偏于乐观化,所指的正是估计偏差。要使 $\text{error}_S(h)$ 对 $\text{error}_D(h)$ 无偏估计,假设 h 和样本 S 必须独立选取。其次,估计偏差这一概念不能与学习器的归纳偏置(inductive bias)相混淆。估计偏差为一数字量,而归纳偏置为一个断言集合。估计量的另一重要属性是它的方差。给定多个无偏估计量,直观上应选取其中方差最小的。由方差的定义,所选择的应为参数值和估计值之间期望平方误差最小的。

假如在测试一假设时,它对 $n=40$ 个随机样例的样本产生 $r=12$ 个错误,那么对 $\text{error}_D(h)$ 的无偏估计为 $\text{error}_S(h) = r/n = 0.3$ 。估计中产生的方差完全来源于 r 中的方差,因为 n 为一常数。由于 r 是二项分布,它的方差由式(2-178)和式(2-179)得 $np(1-p)$ 。然而 p 未知,我们可以用估计量 r/n 来代替 p 。由此得出 r 的估计方差为 $40 \times 0.3 \times (1-0.3) = 8.4$,或相应的标准差 $\sqrt{8.4} \approx 2.9$ 。这表示 $\text{error}_S(h) = r/n$ 中的标准差约为 $2.9/40 = 0.07$ 。总而言之,观察到的 $\text{error}_S(h)$ 为 0.3,标准差约为 0.07。

一般来说,若在 n 个随机选取的样本中有 r 个错误, $\text{error}_S(h)$ 的标准差为

$$\sigma_{\text{error}_S}(h) = \frac{\sigma_r}{n} = \sqrt{\frac{p(1-p)}{n}} \quad (2-183)$$

它约等于用 $r/n = \text{error}_S(h)$ 来代替 p :

$$\sigma_{error_s}(h) \approx \sqrt{\frac{error_s(h)(1-error_s(h))}{n}} \quad (2-184)$$

5. 置信区间

通常描述某估计的不确定性的方法是使用置信区间,真实的值以一定的概率落入该区间中。这样的估计称为置信区间估计。定义:某个参数 p 的 $N\%$ 置信区间是一个以 $N\%$ 的概率包含 p 的区间。

例如,如果在 $n=40$ 个独立抽取的样例的样本中有 $r=12$ 个错误,可以称区间 0.3 ± 0.14 有 95% 的可能性包含真实错误率 $error_D(h)$ 。

如何获得 $error_D(h)$ 的置信区间? 答案在于估计量 $error_s(h)$ 服从二项分布。这一分布的均值为 $error_D(h)$, 标准差可由式(2-184)计算。因此,为计算 95% 置信区间,只需要找到一个以均值 $error_D(h)$ 为中心的区间,它的宽度足以包含该分布下全部概率的 95%。这提供了一个包围 $error_D(h)$ 的区间,使 $error_s(h)$ 必定有 95% 的机会落入其中。同样,它也指定了 $error_D(h)$ 有 95% 的机会落入包围 $error_s(h)$ 的区间的大小。

对于给定的 N 值,如何计算区间大小以使其包含的 $N\%$ 概率质量? 对于二项分布来说这一计算十分烦琐。然而多数情况下可以找到一近似,使计算过程更容易。这基于如下事实,即对于足够大的样本,二项分布可以很好地由正态分布来近似。正态分布是统计学中研究得最透彻的概率分布之一。如图 2-23 所示,正态分布是一个钟形分布,由其均值 μ 和标准差 σ 完全定义。对于大的 n ,二项分布非常近似于一个同样均值和方差的正态分布。

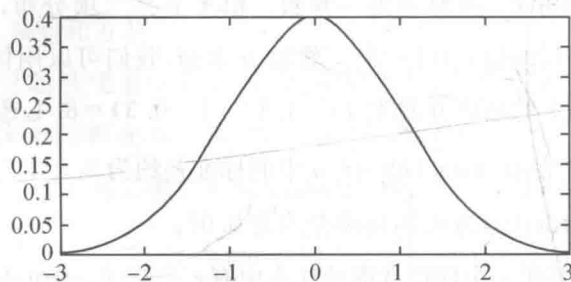


图 2-23 正态分布

一个正态分布(也称为高斯分布)是一个钟形分布。它定义为下面的概率密度函数:

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \quad (2-185)$$

一个正态分布由上面公式中的两个参数 μ 和 σ 完全确定。

如果随机变量 X 遵循正态分布, 则:

(1) X 落入到 (a, b) 的概率为 $\int_a^b p(x) dx$;

(2) X 的期望值或均值 $E(X) = \mu$;

(3) X 的方差 $\text{Var}(X) = \sigma^2$;

(4) X 的标准差 $\sigma_X = \sigma$ 。

之所以使用正态分布来代替, 原因之一是多数统计参考都用列表给出了正态分布下关于均值的包含 $N\%$ 的概率质量的区间的大小。这就是计算 $N\%$ 置信区间所需的信息。实际上, 图 2-24 正是这样一个图。图 2-24 中给定的常数 z_N 定义的是在钟形正态分布下, 包含 $N\%$ 概率质量的关于均值的最小区间的宽度。更精确地说, z_N 以标准差给定了区间的半宽度(即在任一方向距均值的距离), 图 2-24a 给出了针对 $z_{0.80}$ 的一个区间。

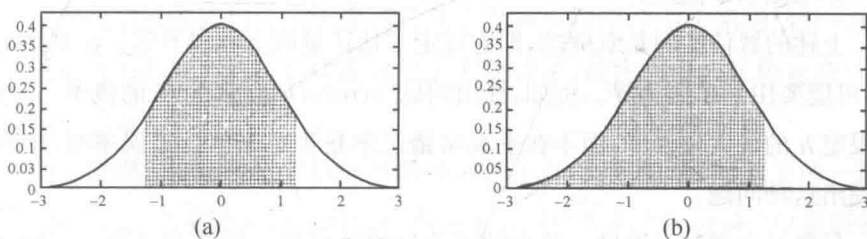


图 2-24 均值为 0, 标准差为 1 的正态分布图

(a) 在 80% 置信度下, 随机变量值位于双侧区间 $[-1.28, 1.28]$ 之间。注意 $z_{0.80} = 1.28$, 有 10% 置信度其落入区间左侧, 10% 落入区间右侧。(b) 在 90% 置信度下, 随机变量位于单侧区间 $[-\infty, 1.28]$ 上。

总而言之, 如果随机变量 Y 服从均值为 μ , 标准差为 σ 的一个正态分布, 那么 Y 的任一观察值 y 有 $N\%$ 的机会落入下面的区间:

$$\mu \pm z_N \sigma \quad (2-186)$$

相似地, 均值 μ 有 $N\%$ 的机会落入下面的区间:

$$y \pm z_N \sigma \quad (2-187)$$

很容易将此结论和前面的结论结合起来,推导式(2-169)的离散值假设的 $N\%$ 置信区间的一般表达式。首先,由于 $\text{error}_S(h)$ 遵从二项分布,其均值为 $\text{error}_D(h)$,标准差如式(2-184)所示;其次,我们知道对于足够大的样本 n ,二项分布非常近似于正态分布;再次式(2-187)告诉我们如何为估计正态分布的均值求出 $N\%$ 置信区间。因此,将 $\text{error}_S(h)$ 的均值和标准差代入到式(2-187)中将得到式(2-169)中对离散值假设的 $N\%$ 置信区间为

$$\text{error}_S(h) \pm z_N \sqrt{\frac{\text{error}_S(h)(1-\text{error}_S(h))}{n}} \quad (2-188)$$

回忆一下,在表达式的推导中有两个近似。

(1)估计 $\text{error}_S(h)$ 的标准差 σ 时,我们将 $\text{error}_D(h)$ 近似为 $\text{error}_S(h)$ (如从式(2-183)到式(2-184)的推导)。

(2)用正态分布近似二项分布。

统计学中的一般规则表明,这两个近似在 $n \geq 30$ 或 $np(1-p) \geq 5$ 时工作得很好。对于较小的 n 值,最好使用列表的形式给出二项分布的具体值。

6. 双侧和单侧边界

上述的置信区间是双侧的,即它规定了估计量的上界和下界。在某些情况下,可能要用到单侧边界。例如,提出问题“ $\text{error}_D(h)$ 至多为 U 的概率”。在只要限定 h 的最大错误率,而不在乎真实错误率是否小于估计错误率时,很自然会提出这种问题。

只要对上述的过程做一些小的修改就可计算单侧错误率边界。它所基于的事实为正态分布关于其均值对称。因此,任意正态分布上的双侧置信区间能够转换为相应的单侧区间,置信度为原来的两倍(见图 2-24(b))。换言之,由一个有下界 L 和上界 U 的 $100(1-\alpha)\%$ 置信区间,可得到一个下界为 L 且无上界的 $100(1-\alpha/2)\%$ 置信区间,同时也可得出一个有上界 U 且无下界的 $100(1-\alpha/2)\%$ 置信区间。这里 α 对应于真实值落在指定区间外的概率。换句话说, α 是真实值落入图 2-24(a)中无阴影部分的概率, $\alpha/2$ 是落入图 2-24(b)的无阴影部分的概率。

为说明这一点,再次考虑 h 产生 $r=12$ 个错误且样本大小 $n=40$ 的这个例子。如上所述,它导致一个双侧的 95% 置信区间 0.3 ± 0.14 。其中 $100(1-$

$\alpha\% = 95\%$, 所以 $\alpha = 0.05$ 。因此, 应用以上规则, 可得有 $100(1 - \alpha/2)\% = 97.5\%$ 的置信度 $\text{error}_D(h)$ 最多为 $0.3 \pm 0.14 = 0.44$, 而不管 $\text{error}_S(h)$ 的下界。因此在 $\text{error}_D(h)$ 上的单侧错误率边界比相应的双侧边界有双倍的置信度。

2.14.4 推导置信区间的一般方法

前面介绍的是针对特定情况推导置信区间估计: 基于独立抽取的 n 个样本, 估计离散值假设的 $\text{error}_D(h)$ 。本节介绍的方法是在许多估计问题中用到的通用的方法。确切地讲, 我们可以将此看作是基于大小为 n 的随机抽取样本的均值来估计总体均值的问题。通用的过程包含以下步骤。

(1) 确定基准总体中要估计的参数 p , 例如 $\text{error}_D(h)$ 。

(2) 定义一个估计量 Y (如 $\text{error}_S(h)$), 它的选择应为最小方差的无偏估计量。

(3) 确定控制估计量 Y 的概率分布 D_y , 包括其均值和方差。

(4) 通过寻找阈值 L 和 U 确定 $N\%$ 置信区间, 以使这个按 D_y 分布的随机变量有 $N\%$ 机会落入 L 和 U 之间。

后面我们将该通用的方法应用到其他几种机器学习中常见的估计问题中去。首先我们需要讨论估计理论的一个重要成果, 被称为中心极限定理 (Central Limit Theorem)。

中心极限定理是简化置信区间推导的一个基本报据。考虑如下的一般框架: 在 n 个独立抽取的且服从同样概率分布的随机变量 $Y_1, \dots, Y_n$ 中观察实验值 (如一硬币的 n 次抛掷)。令 μ 代表每一变量 Y_i 服从的未知分布的均值, 并令 σ 代表标准差, 称这些变量 Y_i 为独立同分布随机变量, 因为它们描述的是各自独立并且服从同样概率分布的实验。为估计 Y_i 服从的分布的均值 μ , 我们计算样本的均值 $\bar{Y}_n = \frac{1}{n} \sum_{i=1}^n Y_i$ (如 n 次投掷硬币中出现正面的比例)。中心极限定理说明在 $n \rightarrow \infty$ 时 $\bar{Y}_n$ 所服从的概率分布为一正态分布, 而不论 Y_i 本身服从什么样的分布。更进一步讲, $\bar{Y}_n$ 服从的分布均值为 μ 而且标准差为 $\frac{\sigma}{\sqrt{n}}$, 精确的定义如下。

定理: 中心极限定理考虑独立同分布的随机变量 $Y_1, \dots, Y_n$ 的集合, 它们

服从一任意的概率分布,均值为 μ ,有限方差为 σ^2 。定义样本均值为 $\bar{Y}$ 而且标准差为 $\frac{\sigma}{\sqrt{n}}$,精确的定义如下:

$$\frac{\bar{Y}_n - \mu}{\frac{\sigma}{\sqrt{n}}} \quad (2-189)$$

它说明在不知道独立的 Y_i 所服从的基准分布的情况下,我们可以得知样本均值 $\bar{Y}$ 的分布形式。更进一步说,中心极限定理说明了怎样使用 $\bar{Y}$ 的均值和方差来确定独立的 Y_i 的均值和方差。

中心极限定理是一个非常有用的结论,因为它表示任意样本均值的估计量(如 $\text{error}_S(h)$)为平均错误率)服从的分布在 n 足够大时可近似为正态分布。如果还知道这一近似的正态分布的方差,就可用式(2-187)来计算置信区间。一个通常的规则是在 $n \geq 30$ 时可使用这一近似。前面的章节我们正是使用了正态分布来近似地描述 $\text{error}_S(h)$ 服从的二项分布。

2.14.5 两个假设错误率间的差异

现在考虑对某离散目标函数有两个假设 h_1 和 h_2 。假设 h_1 在一拥有 n_1 个随机抽取的样例的样本 S_1 上测试,且 h_2 在拥有 n_2 个从相同分布中抽取的样例的样本 S_2 上测试。假定要估计这两个假设的真实错误率间的差异:

$$d \equiv \text{error}_D(h_1) - \text{error}_D(h_2) \quad (2-190)$$

在确定 d 为待估计的参数后,下面要定义一估计量。显然,这里可选择样本错误率之间的差异作为估计量,标记为 $\hat{d}$:

$$\hat{d} \equiv \text{error}_{S_1}(h_1) - \text{error}_{S_2}(h_2) \quad (2-191)$$

在此虽不加证明,但可以认为 $\hat{d}$ 即为 d 的无偏估计量,即 $E[\hat{d}] = d$ 。

随机变量 $\hat{d}$ 服从的概率分布是什么? 从前面的章节中,我们知道对于较大的 n_1 和 n_2 (比如都 ≥ 30), $\text{error}_{S_1}(h_1)$ 和 $\text{error}_{S_2}(h_2)$ 都近似遵从正态分布。由于两正态分布的差仍为一正态分布,因此 $\hat{d}$ 也近似遵从正态分布,均值为 d 。同时,可得出该分布的方差为 $\text{error}_{S_1}(h_1)$ 和 $\text{error}_{S_2}(h_2)$ 的方差的和。使用式(2-184)获得这两个分布的近似方差:

$$\sigma_d^2 \approx \frac{\text{error}_{S_1}(h_1)(1 - \text{error}_{S_1}(h_1))}{n_1} + \frac{\text{error}_{S_2}(h_2)(1 - \text{error}_{S_2}(h_2))}{n_2} \quad (2-192)$$

现在已确定了估计量 $\hat{d}$ 所服从的概率分布,很容易推导出置信区间以说明使用 $\hat{d}$ 来估计 d 的可能误差。随机变量 $\hat{d}$ 服从均值 d 方差 σ^2 的正态分布,其 $N\%$ 置信区间估计为 $\hat{d} \pm z_N \sigma$ 。使用上面给出的方差 σ_d^2 的近似值, d 的近似的 $N\%$ 置信区间估计为

$$\hat{d} \pm z_N \sqrt{\frac{\text{error}_{S_1}(h_1)(1 - \text{error}_{S_1}(h_1))}{n_1} + \frac{\text{error}_{S_2}(h_2)(1 - \text{error}_{S_2}(h_2))}{n_2}} \quad (2-193)$$

其中, z_N 是常数。上式给出了一般的双侧置信区间,以估计两个假设错误率之间的差异。有时可能需要某一置信度下的单侧的边界——要么界定最大可能差异,要么为最小的可能差异。单侧置信区间可以通过第 2.14.3 节中描述的方法修改上式而得到。虽然上面的分析考虑的是 h_1 和 h_2 在相互独立的数据样本上测试的情况,但是在一个样本 S (S 仍然独立于 h_1 和 h_2) 上测试 h_1 和 h_2 并用公式(2-193)计算置信区间通常也是可接受的。这样, $\hat{d}$ 重新定义为

$$\hat{d} \equiv \text{error}_S(h_1) - \text{error}_S(h_2) \quad (2-194)$$

当使用 S 来代替 S_1 和 S_2 时,新的 $\hat{d}$ 中的方差通常小于式(2-192)给出的方差。这是因为使用单个的样本 S 消除了由 S_1 和 S_2 的组合带来的随机差异。这样,由式(2-193)给出的置信区间一般说来会过于保守,但仍然是正确的。

【假设检验】:有时我们感兴趣的是某个特定的猜想正确的概率,而不是对某参数的置信区间估计。比如下面的问题,“ $\text{error}_D(h_1) > \text{error}_D(h_2)$ 的可能性有多大?”。仍使用前一节的条件设定,假定要测量 h_1 和 h_2 的样本错误率,使用大小为 100 的独立样本 S_1 和 S_2 ,并且知道 $\text{error}_{S_1}(h_1) = 0.30$ 且 $\text{error}_{S_2}(h_2) = 0.20$,因此差异 $\hat{d}$ 为 0.10。当然,由于数据样本的随机性,即使 $\text{error}_D(h_1) \leq \text{error}_D(h_2)$,仍有可能得到这样的差异。在这里,给定 $\hat{d} = 0.10$, $\text{error}_D(h_1) > \text{error}_D(h_2)$ 的概率是多少? 与此相对应,如何计算在 $\hat{d} = 0.10$ 时, $d > 0$ 的概率?

注意,概率 $Pr(d > 0)$ 等于 $\hat{d}$ 对 d 的过高估计不大于 0.1 的概率,也就是这个概率为 $\hat{d}$ 落入单侧区间 $\hat{d} < d + 0.10$ 的概率。由于 d 是 $\hat{d}$ 所服从的分布的

均值,上式等价于 $\hat{d} < \mu_d + 0.10$ 。概括地说,概率 $Pr(d > 0)$ 等于 $\hat{d}$ 落入单侧区间 $\hat{d} < \mu_d + 0.10$ 概率。由于前面我们已计算出 $\hat{d}$ 的大致分布,就可以通过 $\hat{d}$ 分布在该区间的概率质量来确定 $\hat{d}$ 落入这个单侧区间的概率。首先,将区间 $\hat{d} < \mu_d + 0.10$ 用允许偏离均值的标准差的数目来重新表示。使用式(2-192)可得, $\sigma_d \approx 0.061$,所以这一区间可近似表示为

$$\hat{d} < \mu_d + 1.64\sigma_d \quad (2-195)$$

与此正态分布的单侧区间相关联的置信度是多少呢?查表 2-2 可得,关于均值的 1.64 标准差对应置信度 90% 的双侧区间。因此这个单侧区间具有 95% 的置信度。因此,给定观察到的 $\hat{d} = 0.1$, $\text{error}_D(h_1) > \text{error}_D(h_2)$ 的概率约为 0.95。

根据统计学的术语可表述为:接受(accept)“ $\text{error}_D(h_1) > \text{error}_D(h_2)$ ”这一假设,置信度为 0.95。换一种说法可表述为:我们拒绝(reject)对立假设(常称为零假设),以 $(1 - 0.95) = 0.05$ 的显著水平(significance level)。

2.14.6 学习算法比较

有时我们更感兴趣的是比较两个学习算法 L_A 和 L_B 的性能,而不是两个具体的假设本身。怎样近似地检验多个学习算法,如何确定两个算法之间的差异在统计上是有意义的?虽然,在机器学习研究领域,关于学习算法比较哪个方法最好仍存在激烈的争论,不过这里介绍了一个合理的途径。

通常,先指定要估计的参数。假定有 L_A 和 L_B 两个算法,要确定为了学习一个特定目标函数 f ,需通过平均来说那个算法更好。定义“平均”的一种合理方法是,从一基准实例分布 D 中拍取包含 n 个样例的训练集合,在所有这样的集合中测量两个算法的平均性能。换句话说,需要估计假设错误率之间差异的期望值:

$$E_{S \sim D} [\text{error}_D(L_A(S)) - \text{error}_D(L_B(S))] \quad (2-196)$$

其中, $L(S)$ 代表在给定训练数据的样本 S 上,学习算法 L 输出的假设,下标 $S \sim D$ 表示期望值是在基准分布 D 中抽取的样本 S 上计算。上述表达式描述的是学习算法 L_A 和 L_B 之间的差异的期望值。

在实际的学习算法比较中,我们只有一个有限的样本 D_0 。在这种情况下,

很显然,要估计上述的量需要将 D_0 分割成训练集合 S_0 和与之不相交的测试集合 T_0 。训练数据可以用来既训练 L_A 又训练 L_B ,而测试数据则用来比较两个学习到的假设的准确度,也就是使用下式来计算:

$$\text{error}_{T_0}(L_A(S_0)) - \text{error}_{T_0}(L_B(S_0)) \quad (2-197)$$

上式与式(2-196)的计算有两个重要的不同。首先我们使用 $\text{error}_{T_0}(h)$ 来近似 $\text{error}_D(h)$;其次,错误率的差异测量是在一个训练集合 S_0 上,而不是在从分布 D 中抽取的所有的样本 S 上计算期望值。

改进式(2-197)的一种方法是将数 D_0 多次分割为不相交的训练和测试集合,然后在其中计算这些不同的试验的错误率的平均值。它在一可用的固定数据样本 D_0 上估计两个学习算法错误率之间的差异。该过程首先将数据拆分为 k 个不相交的相等子集,子集大小至少为 30,然后训练和测试算法 k 次,每次使用其中一个子集作为测试数据集,其他 $k-1$ 个子集为训练集。使用这种办法,学习算法在 k 个独立测试集上测试,而把错误率的差异的均值 $\bar{\delta}$ 作为两个学习算法间差异的估计。

估计两个学习算法 L_A 和 L_B 错误率差异的过程。

(1)将可用数据 D_0 分割成 k 个相同大小的不想交子集 $T_1, T_2, \dots, T_k$ 。其大小至少 30。

(2)令 i 从 1 到 k 循环,做下面的操作:

使用 T_i 作为测试集合,而剩余的数据作为训练集合 S_i ;

$S_i \leftarrow |D_0 - T_i|$; $h_A \leftarrow L_A(S_i)$; $h_B \leftarrow L_B(S_i)$; $\delta_i \leftarrow \text{error}_{T_i}(h_A) - \text{error}_{T_i}(h_B)$ 。

(3)返回值 $\bar{\delta}$,其中:

$$\bar{\delta} = \frac{1}{k} \sum_{i=1}^k \delta_i$$

返回的 $\bar{\delta}$ 可被用作对公式(2-196)所需的量的一个估计。更合适的说法是把 $\bar{\delta}$ 看作是下式的估计:

$$E_{s \subset D_0} [\text{error}_D(L_A(S)) - \text{error}_D(L_B(S))] \quad (2-198)$$

其中, S 代表一个大小为 $\frac{k-1}{k} |D_0|$ 且从 D_0 中均匀抽取出的随机样本。在该式和式(2-196)中最初的表达式之间,唯一的差别在于其期望值的计算是在可用

数据 D_0 的子集上计算,而不是在从整个分布 D 上抽取的子集上计算。

估计式(2-198)的近似的 $N\%$ 置信区间,可用 $\bar{\delta}$ 表示为

$$\bar{\delta} \pm t_{N,k-1} s_{\bar{\delta}} \quad (2-199)$$

其中, $t_{N,k-1}$ 是一常数,其意义类似于前面置信区间表达式中的 z_N ,而 $s_{\bar{\delta}}$ 代表对 $\bar{\delta}$ 多所服从的概率分布的标准差的估计,更确切地讲, $s_{\bar{\delta}}$ 的定义为

$$s_{\bar{\delta}} = \sqrt{\frac{1}{k(k-1)} \sum_{i=1}^k (\delta_i - \bar{\delta})^2} \quad (2-200)$$

注意,式(2-199)中的常量 $t_{N,k-1}$ 有两个下标。第一个代表所需的置信度,如前面的常数 z_N 中那样。第二个参数称为自由度(degree of freedom),常被记作 v ,它与生成随机变量 δ 的值时独立的随机事件数目相关。在当前的条件下,自由度数值为 $k-1$ 。参数 t 的几种取值在表 2-3 中列出。注意当 $k \rightarrow \infty$ 时, $t_{N,k-1}$ 的值趋向于常数 z_N 。

表 2-3 双侧置信区间 $t_{N,v}$ 的值

自由度	置信度 N			
	90%	95%	98%	99%
$v=2$	2.92	4.30	6.96	9.92
$v=5$	2.02	2.57	3.36	4.03
$v=10$	1.81	2.23	2.76	3.17
$v=20$	1.72	2.09	2.53	2.84
$v=30$	1.70	2.04	2.46	2.75
$v=120$	1.66	2.98	2.36	2.62
$v=\infty$	1.64	1.96	2.33	2.58

(注:当 $v \rightarrow \infty$ 时, $t_{N,v}$ 趋近于 z_N 。)

值得注意的是这里描述的比较学习算法的过程要在同样的测试集合上测试两个假设。这与 2.14.5 节中描述的比较两个用独立测试集合评估过的假设不同。使用相同样本来测试假设被称为配对测试。配对测试通常会产生更紧密的置信区间。因为在配对测试中任意的差异都来源于假设之间的差异。相反,若假设在分开的数据样本上的测试,两个样本错误率之间的差异也可能部分来源于两个样本组成上的不同。

1. 配对 t 测试

上面描述了在给定固定数据集时比较两个学习算法的过程。如果第一次阅读,为不失连续性可以跳过它。

为了解释置信区间,考虑以下的估计问题:(1)给定一系列独立同分布的随机变量 $Y_1, Y_2, \dots, Y_k$ 的观察值;(2)要估计这些 Y_i 所服从的概率分布的均值 μ ;(3)使用的估计量为样本均值 $\bar{Y}$ 。

基于样本均值 $\bar{Y}$ 估计分布均值 μ 的问题非常普遍。例如,它覆盖了早先用 $\text{error}_S(h)$ 来估计 $\text{error}_D(h)$ 的问题(其中, Y_i 为 0 或 1 表示 h 是否对一单独的 S 样例产生误分类,而 $\text{error}_D(h)$ 为基准分布的均值 μ)。由式(2-198)和式(2-199)描述的 t 测试应用于该问题的一特殊情形——即每个单独的 Y_i 都遵循正态分布。

现考虑表 2-3 比较学习算法的过程的一个理想化形式。假定不是拥有固定样本数据 D_0 ,而是从基准实例分布中抽取新的训练样例。在这里我们修改过程,使每一次循环生成新的随机训练集 S_i 和新的随机测试集 T_i ,生成方法是从基准实例分布中抽取而不是从固定样本 D_0 中抽取。这一理想化方法能很好地匹配上面的估计问题,特别是该过程所测量的 δ_i ,对应到独立同分布的随机变量 Y_i ,其分布的均值 μ 对应两学习算法错误率的期望差异(如式(2-196),样本均值 $\bar{Y}$ 为这一理想化方法计算出的 $\bar{\delta}$ 。我们希望回答“ $\bar{\delta}$ 是否能较好地估计 μ ”这一问题。

首先注意到,测试集 T_i 至少包含 30 个样例。因此,单独的 δ_i 将近似遵循正态分布(由中心极限定理可知)。因此,我们有一特殊条件,即 Y_i 服从近似的正态分布。我们知道,一般当每个 Y_i 遵循正态分布时,样本均值 $\bar{Y}$ 也遵循正态分布。由此,可以考虑使用前面计算置信区间的表达式,其中的估计量正遵循正态分布。然而,该公式要求我们知道这个分布的标准差,但现在标准差未知。

t 测试正好用于这样的情形,即估计一系列独立同正态分布的随机变量的样本均值。在这里,可使用式(2-198)和式(2-199)中的置信区间,它可被重新表述为

$$\mu = \bar{Y} \pm t_{N,k-1} s_Y \quad (2-201)$$

其中, s_Y 为估计的样本均值的标准差:

$$s_Y = \sqrt{\frac{1}{k(k-1)} \sum_{i=1}^k (Y_i - \bar{Y})^2} \quad (2-202)$$

这里, $t_{N,k-1}$ 是一个类似于前面的 z_N 的常量。实际上, 常量 $t_{N,k-1}$ 描述的是被称为 t 分布的概率分布下的区域, 正如常数 z_N 描述了正态分布下的区域。 t 分布是一类似于正态分布的钟形分布, 但更宽且更矮, 以反映由于使用 s_Y 来近似真实的标准差 σ_Y 时带来的更大的方差。当 k 趋近于无穷时, t 分布趋近于正态分布 (因此, $t_{N,k-1}$ 趋近于 z_N)。这在直觉上是正确的, 因为我们希望样本规模 k 增加时 s_Y 收敛到真实的标准差 σ_Y , 并且当标准差确切已知时可使用 z_N 。

2. 实际考虑

上面的讨论证明了在使用样本均值来估计一个包含 k 个独立同正态分布的随机变量的样本均值时, 可使用式 (2-199) 来估计置信区间。该结论在我们的理想化方法中是合适的, 即假定对于目标函数的样例可进行无限存取。在实际中, 若数据集 D_0 有限, 且算法使用之前描述的实际方法, 其实这一证明并不严格适用。实际的问题是, 为产生 δ_i 只有重新采样 D_0 , 用不同的方法把它分割为测试集和训练集。此时 δ_i 相互之间并不独立, 因为它们基于从有限子集 D_0 中抽取的相互重叠的训练样例。而不是从整个分布 D 中抽取的样例。

当只有一个有限的数据样本 D_0 可用时, 有几种方法可被用来重采样 D_0 。表 2-3 描述的是 k -fold 方法, 其中 D_0 被分为 k 个不相交的等大小的子集。在这种 k -fold 方法中, D_0 中每一样例都有一次用于测试, 而 $k-1$ 次用于训练。另一种常用的方法是从 D_0 中随机抽取至少有 30 个样例的测试集合, 再用剩余的样例来训练, 重复这一过程直到足够的次数。这种随机方法的好处是能够重复无数次, 以减小置信区间到需要的宽度。相反, k -fold 方法受限于样例的总数, 这是因为每个样例只有一次用于测试, 且希望样本大小至少为 30。然而, 随机方法的缺点是, 测试集合不能再被看作是从基准实例分布 D 中独立抽取。相反, k -fold 交叉验证生成的测试集合是独立的, 因为一个实例只在测试集合中出现一次。

概括地说, 基于有限数据的学习算法的比较中没有一个单独的方法能满足我们希望的所有约束。有必要记住, 统计学模型在数据有限时很少能完美地匹配学习算法验证中的所有约束。然而, 它们确实提供了近似的置信区间, 有助于解释学习算法的实验性比较。

第3章 人工神经网络

人类具有比其他生物更为发达的大脑,大脑是任何思维活动的物质基础,集中体现了人类的智慧。自古以来,从事脑研究的科学家想方设法了解和揭示大脑的工作原理和思维活动的机理,人工智能研究的科学家则对如何构造出具有人类智能的人工智能系统,以模拟、延伸和扩展脑功能,完成类似于人脑的工作非常感兴趣。而后,诞生了人工神经网络技术,这为人类进一步研究了解人脑思维的奥秘,仿生人类的大脑活动指明了新的方向。

人类脑的思维活动包括逻辑思维和形象思维两种模式。前者的基础有三项:概念、判断与推理。它们的过程是:第一步是将信息抽象为概念;第二步是判断;第三步是再根据已有的规则进行逻辑推理。我们可以用符号来表示概念的含义,逻辑推理则可以以串行方式进行,所以,我们可以使用计算机语言描述这个过程,使它按照串行的指令,进而交由机器来完成。自20世纪40年代第一台电子计算机的问世,它的本质就是使用机器模拟人脑逻辑思维运行模式,这样的系统称之为人工智能系统。毋庸置疑的,现代计算机的计算速度比人脑的计算速度高上千万倍,计算机程序对于那些有明确的特征,运算起来有条理的逻辑问题,计算机能够高速高效地完成解答,人类大脑的能力远远不及计算机在数值运算和逻辑运算方面的精确与高速。

人们研究人工神经网络已有几十年的历史。需要特别指出的有几个事件,首先,20世纪最后几年,在模拟人脑逻辑思维方面,计算机模拟人脑逻辑思维与人类象棋高手进行了两场举世瞩目的人机大战。1996年2月,由美国IBM公司开发的名为“深蓝”^①计算机与当时的国际象棋世界冠军斯帕罗夫进行了总共为6局的对弈,最终,世界冠军以4:2的比分战胜了他的计算机对手,当时深蓝的计算能力是每秒1亿步。时隔一年的1997年,卡斯帕罗夫再次与计

^①“深蓝”是美国IBM公司生产的一台超级国际象棋电脑,深蓝在世界超级电脑中排名第259位,计算能力为每秒113.8亿次浮点运算。1997年的深蓝可分析及估计对手的12步棋,而一名人类象棋好手最多可估计10步棋。

算能力已提高到每秒 2 亿步的“深蓝”第二代交战时,最终却以 2.5 : 3.5 的比分惜败。2006 年 8 月,令人关注的是一场中国象棋界 5 位顶尖棋手组成的棋手联队与当时的超级计算机“浪潮天梭”^①之间的人机大战,经过十一局的苦战,机器队最终以 5.5 : 4.5 的总比分战胜棋手联队。最新的这一场“人机大战”或“人工智能 PK 人类智慧”发生在 2016 年 3 月,这是在与韩国围棋九段名将李世石的对弈中,“阿尔法围棋”(AlphaGo)^②机器最终以 4 : 1 取得的战绩。

3.1 概述

3.1.1 什么是人工神经网络

迄今为止的各代计算机都是基于冯·诺依曼工作原理:其信息存储与处理是分开的,即存储器与处理器相互独立;处理的信息必须是形式化信息,即用二进制编码定义的文字、符号、数字、指令和各种规范化的数据格式、命令格式等;而信息处理的方式必须是串行的,即 CPU 不断地重复取址、译码、执行、存储这 4 个步骤。这种计算机的结构和串行工作方式决定了它只擅长于数值和逻辑运算。

人类的大脑大约有 1.4×10^{11} 个神经细胞,亦称为神经元。每个神经元有数以千计的通道同其他神经元广泛地相互连接,形成复杂的生物神经网络。生物神经网络以神经元为基本信息处理单元,对信息进行分布式存储与加工,这种信息加工与存储相结合的群体协同工作方式使得人脑呈现出目前计算机无法模拟的神奇智能。为了进一步模拟人脑的形象思维方式,人们不得不跳出冯·诺依曼计算机的框架另辟蹊径。而从模拟人脑生物神经网络的信息存储、加工处理机制入手,设计具有人类思维特点的智能机器,无疑是最有希望的途径之一。

用计算方法对神经网络信息处理规律进行探索称为计算神经科学,该方法对于阐明人脑的工作原理具有深远意义。人脑的信息处理机制是在漫长的进

①“浪潮天梭”具有平均每步棋 27 秒的速度,每步 66 万亿次的棋位分析与检索能力。

②阿尔法围棋(AlphaGo)是一款围棋人工智能程序,由谷歌(Google)旗下 DeepMind 公司的戴密斯·哈萨比斯、大卫·席尔瓦、黄士杰和与他们的团队开发。其主要工作原理是“深度学习”。

化过程中形成和完善的。虽然近年来,在细胞和分子水平上对脑结构和脑功能的研究已经有了长足的发展。然而到目前为止,人类对神经系统内的电信号和化学信号是怎样被用来处理信息的只有模糊的概念。尽管如此,把通过分子和细胞水平的技术所达到的微观层次与通过行为研究达到的系统层次结合起来,可以形成对人脑神经网络的基本认识。在此基本认识的基础上,以数学和物理方法以及信息处理的角度对人脑神经网络进行抽象,并建立某种简化模型,就称为人工神经网络(Artificial Neural Network, ANN)。人工神经网络远不是人脑生物神经网络的真实写照,而只是对它的简化、抽象与模拟。揭示人脑的奥妙不仅需要各学科的交叉和各领域专家的协作,还需要测试手段的进一步发展。尽管如此,目前已提出上百种人工神经网络模型。令人欣慰的是,这种简化模型的确能反映出人脑的许多基本特性,如自适应性、自组织性和很强的学习能力。它们在模式识别、系统辨识、自然语言理解、智能机器人、信号处理、自动控制、组合优化、预测预估、故障诊断、医学与经济学等领域已成功地解决了许多现代计算机难以解决的实际问题,表现出良好的智能特性。

目前,关于人工神经网络的定义尚不统一,例如,美国神经网络学家 Hecht Hielsen 关于人工神经网络的一般定义是“神经网络是由多个非常简单的处理单元彼此按某种方式相互连接而形成的计算系统,该系统是靠其状态对外部输入信息的动态响应来处理信息的”。美国国防高级研究计划局关于人工神经网络的解释是:“人工神经网络是一个由许多简单的并行工作的处理单元组成的系统,其功能取决于网络的结构、连接强度以及各单元的处理方式。”综合人工神经网络的来源、特点及各种解释,可以简单表述为:人工神经网络是一种旨在模仿人脑结构及其功能的脑式智能信息处理系统。为叙述简便,后面常将人工神经网络简称为神经网络。

3.1.2 神经网络的基本特点与功能

人工神经网络是基于对人脑组织结构、活动机制的初步认识提出的一种新型信息处理体系。通过模仿脑神经系统的组织结构以及某些活动机理,人工神经网络可呈现出人脑的许多特征,并具有人脑的一些基本功能。

1. 神经网络的基本特点

这里从结构、性能和能力 3 个方面介绍神经网络的基本特点。

(1)结构特点:信息处理的并行性、信息存储的分布性、信息处理单元的互连性、结构的可塑性。

人工神经网络是由大量简单处理元件相互连接构成的高度并行的非线性系统,具有大规模并行性处理特征。虽然每个处理单元的功能十分简单,但大量简单处理单元的并行活动使网络呈现出丰富的功能并具有较快的速度。结构上的并行性使神经网络的信息存储必然采用分布式方式,即信息不是存储在网络的某个局部,而是分布在网络所有的连接权中。一个神经网络可存储多种信息,其中每个神经元的连接权中存储的是多种信息的一部分。当需要获得已存储的知识时,神经网络在输入信息激励下采用“联想”的办法进行回忆,因而具有联想记忆功能。神经网络内在的并行性与分布性表现在其信息的存储与处理都是空间上分布、时间上并行的。

(2)性能特点:高度的非线性、良好的容错性和计算的非精确性。

神经元的广泛互联与并行工作必然使整个网络呈现出高度的非线性特点。而分布式存储的结构特点会使网络在两个方面表现出良好的容错性:一方面,由于信息的分布式存储,当网络中部分神经元损坏时不会对系统的整体性能造成影响,这一点就像人脑中每天都有神经细胞正常死亡而不会影响大脑的功能一样;另一方面,当输入模糊、残缺或变形的信息时,神经网络能通过联想恢复完整的记忆,从而实现对不完整输入信息的正确识别,这一特点就像人可以对不规范的手写字进行正确识别一样。神经网络能够处理连续的模拟信号以及不精确的、不完全的模糊信息,因此给出的是次优的逼近解而非精确解。

(3)能力特征:自学习、自组织与自适应性。

自适应性是指一个系统能改变自身的性能以适应环境变化的能力,它是神经网络的一个重要特征。自适应性包含自学习与自组织两层含义。神经网络的自学习是指当外界环境发生变化时,经过一段时间的训练或感知,神经网络能通过自动调整网络结构参数,使得对于给定输入能产生期望的输出,训练是神经网络学习的途径,因此经常将学习与训练两个词混用。神经网络能在外部刺激下按一定规则调整神经元之间的突触连接,逐渐构建起神经网络,这一构建过程称为网络的自组织(或称重构)。神经网络的自组织能力与自适应性相关,自适应性是通过自组织实现的。

2. 神经网络的基本功能

人工神经网络是借鉴生物神经网络而发展起来的新型智能信息处理系统,由于其结构上“仿造”了人脑的生物神经系统,因而其功能上也具有了某种智能特点。下面对神经网络的基本功能进行简要介绍。

(1) 联想记忆

由于神经网络具有分布存储信息和并行计算的性能,因此它具有对外界刺激信息和输入模式进行联想记忆的能力。这种能力是通过神经元之间的协同结构以及信息处理的集体行为而实现的。神经网络是通过其突触权值和连接结构来表达信息的记忆,这种分布式存储使得神经网络能存储较多的复杂模式和恢复记忆的信息。神经网络通过预先存储信息和学习机制进行自适应训练,可以从不完整的信息和噪声干扰中恢复原始的完整信息,这一能力使其在图像复原、图像和语音处理、模式识别、分类等方面具有巨大的潜在应用价值。

联想记忆有两种基本形式:自联想记忆与异联想记忆。

① 自联想记忆

网络中预先存储(记忆)多种模式信息,当输入某个已存储模式的部分信息或带有噪声干扰的信息时,网络能通过动态联想过程回忆起该模式的全部信息,如图 3-1(a)所示。

② 异联想记忆

如图 3-1(b)所示,网络中预先存储了多个模式对,每一对模式均由两部分组成,当输入某个模式对的一部分时,即使输入信息是残缺的或叠加了噪声的,网络也能回忆起与其对应的另一部分。

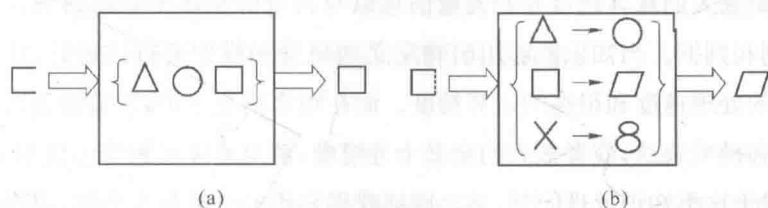


图 3-1 联想记忆

(2) 非线性映射

在客观世界中,许多系统的输入与输出之间存在复杂的非线性关系,对于

这类系统,往往很难用传统的数理方法建立其数学模型。设计合理的神经网络通过对系统输入输出样本对进行自动学习,能够以任意精度逼近任意复杂的非线性映射。神经网络的这一优良性能使其可以作为多维非线性函数的通用数学模型。该模型的表达是非解析的,输入输出数据之间的映射规则由神经网络在学习阶段自动抽取并分布式存储在网络的所有连接中。具有非线性映射功能的神经网络应用十分广阔,几乎涉及所有领域。

(3) 分类与识别

神经网络对外界输入样本具有很强的识别与分类能力。对输入样本的分类实际上是在样本空间找出符合分类要求的分割区域,每个区域内的样本属于一类。传统分类方法只适合解决同类相聚、异类分离的识别与分类问题。但客观世界中许多事物(例如不同的图像、声音、文字等)在样本空间上的区域分割曲面是十分复杂的,相近的样本可能属于不同的类,而远离的样本可能同属一类。神经网络可以很好地解决对非线性曲面的逼近,因此比传统的分类器具有更好的分类与识别能力。

(4) 优化计算

优化计算是指在已知的约束条件下,寻找一组参数组合,使由该组合确定的目标函数达到最小值。某些类型的神经网络可以把待求解问题的可变参数设计为网络的状态,将目标函数设计为网络的能量函数。神经网络经过动态演变过程达到稳定状态时对应的能量函数最小,从而其稳定状态就是问题的最优解。这种优化计算不需要对目标函数求导,其结果是网络自动给出的。

(5) 知识处理

知识是人们从客观世界的大量信息以及自身的实践中总结归纳出来的经验、规则和判据。当知识能够用明确定义的概念和模型进行描述时,计算机具有极快的处理速度和很高的运算精度。而在很多情况下,知识常常无法用明确的概念和模型表达,或者概念的定义十分模糊,甚至解决问题的信息不完整、不全面,对于这类知识处理问题,神经网络获得知识的途径与人类似,也是从对象的输入输出信息中抽取规律而获得关于对象的知识,并将知识分布在网络的连接中予以存储。一方面,神经网络的知识抽取能力使其能够在没有任何先验知识的情况下自动从输入数据中提取特征、发现规律,并通过自组织过程构建网

络,使其适合于表达所发现的规律;另一方面,人的先验知识可以大大提高神经网络的知识处理能力,两者相结合会使神经网络智能得到进一步提升。

3.2 神经网络的应用领域介绍

神经网络的脑式智能信息处理特征与能力使其应用领域日益扩大,潜力日趋明显。许多用传统信息处理方法无法解决的问题采用神经网络后取得了良好的效果。下面介绍一下目前神经网络的几个主要应用领域。

1. 信息处理领域

神经网络作为一种新型智能信息处理系统,其应用贯穿信息的获取、传输、接收与加工利用等各个环节,这里仅列举几个方面的应用。

(1) 信号处理

神经网络广泛应用于自适应信号处理和非线性信号处理中。前者如信号的自适应滤波、时间序列预测、谱估计、噪声消除等;后者如非线性滤波、非线性预测、非线性编码、调制/解调等。在信号处理方面神经网络有着许多成功应用的实例,第一个成功应用的实例就是电话线中回声的消除,其他还有雷达回波的多目标分类、运动目标的速度估计、多探测器的信息融合等。

(2) 模式识别

模式识别涉及模式的预处理变换和将一种模式映射为其他类型的操作,神经网络在这两个方面都有许多成功的应用。神经网络不仅可以处理静态模式如固定图像、固定能谱等,还可以处理动态模式如视频图像、连续语音等。大家所熟悉的静态模式识别的成功例子有手写字的识别,动态模式识别的成功实例有语音信号的识别。目前电脑市场上随处可见的手写输入和语音输入系统进一步表明神经网络在模式识别方面的应用已经商品化。

(3) 数据压缩

在数据传送与存储时,数据压缩至关重要。神经网络可对待传送(或待存储)的数据提取模式特征,只将该特征传出(或存储),接收后(或使用)再将其恢复成原始模式。

2. 自动化领域

20世纪80年代以来,神经网络和控制理论与控制技术相结合,发展为自

动控制领域的一个前沿学科——神经网络控制。它是智能控制的一个重要分支,为解决复杂的非线性、不确定、不确知系统的控制问题开辟了一条新的途径。神经网络用于控制领域,已取得以下主要进展。

(1) 系统辨识

在自动控制问题中,系统辨识的目的是为了建立被控对象的数学模型。多年来控制领域对于复杂的非线性对象的辨识,一直未能很好地解决。神经网络所具有的非线性特性和学习能力,使其在系统辨识方面有很大的潜力,为解决具有复杂的非线性、不确定性和不确知对象的辨识问题开辟了一条有效途径。基于神经网络的系统辨识是以神经网络作为被辨识对象的模型,利用其非线性特性,可建立非线性系统的静态或动态模型。

(2) 神经控制器

由于控制器在实时控制系统中起着“大脑”的作用,神经网络具有自学习和自适应等智能特点,因而非常适合于作控制器。对于复杂非线性系统,神经控制器所达到的控制效果往往明显好于常规控制器。近年来,神经控制器在工业、航空以及机器人等领域的控制系统应用中已取得许多可喜的成就。

(3) 智能检测

所谓智能检测一般包括干扰量的处理、传感器输入输出特性的非线性补偿、零点和量程的自动校正以及自动诊断等。这些智能检测功能可以通过传感元件和信号处理元件的功能集成来实现。随着智能化程度的提高,功能集成型已逐渐发展为功能创新型,如复合检测、特征提取及识别等,而这类信息处理问题正是神经网络的强项。在对综合指标的检测(例如对环境舒适度这类综合指标的检测)中,以神经网络作为智能检测中的信息处理元件便于对多个传感器的相关信息(如温度、湿度、风向和风速等)进行复合、集成、融合、联想等数据融合处理,从而实现单一传感器所不具备的功能。

3. 工程领域

20世纪80年代以来,神经网络的理论研究成果已在众多的工程领域取得了丰硕的应用成果,下面的介绍仅助读者窥其一斑。

(1) 汽车工程

汽车在不同状态参数下运行时,能获得最佳动力性与经济性的挡位称为最

佳挡位。由于神经网络具有良好的非线性映射能力,通过学习优秀驾驶员的换挡经验数据,可自动提取蕴含在其中的最佳换挡规律。神经网络在汽车刹车自动控制系统中也有成功的应用,该系统能在给定刹车距离、车速和最大减速度的情况下,以人体能感受到的最小冲击实现平稳刹车,而不受路面坡度和车重的影响。随着国内外对能源短缺和环境污染问题的日趋关切,燃油消耗率和排烟度愈来愈受到人们的关注。神经网络在载重车柴油机燃烧系统方案优化中的应用,有效地降低了油耗和排烟度,获得了良好的社会效益。

(2) 军事工程

神经网络同红外搜索与跟踪系统配合后可以发现与跟踪飞行器。一个成功的例子是,利用神经网络检测空间卫星的动作状态是稳定、倾斜、旋转还是摇摆,正确率可达95%。利用声呐信号判断水下目标是潜艇还是礁石是军事上常采用的办法。借助神经网络的语音分类与信号处理上的经验对声呐信号进行分析研究,对水下目标的识别率可达90%。密码学研究一直是军事领域中的重要研究课题,利用神经网络的联想记忆特点可设计出密钥分散保管方案;利用神经网络的分类能力可提高密钥的破解难度;利用神经网络还可设计出安全的保密开关,如语音开关、指纹开关等。

(3) 化学工程

20世纪80年代中期以来,神经网络在制药、生物化学、化学工程等领域的研究与应用蓬勃开展,取得了不少成果。例如,在光谱分析方面,应用神经网络在红外光谱、紫外光谱、折射光谱和质谱与化合物的化学结构间建立某种确定的对应关系方面的成功应用实例比比皆是。此外,还有将神经网络用于判定化学反应的生成物;用于判定钾、钙、硝酸、氯等离子的浓度;用于研究生命体中某些化合物的含量与其生物活性的对应关系等大量应用实例。

(4) 水利工程

近年来,我国水利工程领域的科技人员已成功地将神经网络的方法用于水力发电过程辨识和控制、河川径流预测、河流水质分类、水资源规划、混凝土性能预估、拱坝优化设计、预应力混凝土桩基等结构损伤诊断、砂土液化预测、岩体可爆破性分级及爆破效应预测、岩土类型识别、地下工程围岩分类、大坝等工程结构安全监测、工程造价分析等许多实际问题中。

4. 医学领域

(1) 检测数据分析

许多医学检测设备的输出数据都是连续波形的形式,这些波的极性和幅值常常能够提供有意义的诊断依据。神经网络在这方面的应用非常普遍,一个成功的应用实例是用神经网络进行多道脑电棘波的检测。很多癫痫病人常规治疗往往无效,但他们可得益于脑电棘波检测系统。脑电棘波的出现通常意味着脑功能的某些异常,棘波的极性和幅值经常提供了异常的部位和程度信息,因而神经网络脑电棘波检测系统可用来提供脑电棘波的实时检测和癫痫的预报。

(2) 生物活性研究

用神经网络对生物学检测数据进行分析,可提取致癌物的分子结构特征建立分子结构和致癌活性之间的定量关系,并对分子致癌活性进行预测。分子致癌性的神经网络预测具有生物学检测所不具备的优点,它不仅可对新化合物的致癌性和致突变性预先做出评价,从而避免盲目投入造成浪费,而且检测费用低,可作为致癌物大面积预筛的工具。

(3) 医学专家系统

专家系统在医疗诊断方面有许多应用。虽然专家系统的研究与应用取得了重大进展,但由于知识“爆炸”和冯·诺依曼计算机的“瓶颈”问题使其应用受到严重挑战。以非线性并行分布式处理为基础的神经网络为专家系统的研究开辟了新的途径,利用其学习功能、联想记忆功能和分布式并行信息处理功能,来解决专家系统中的知识表示、获取和并行推理等问题取得了良好效果。

5. 经济领域

(1) 信贷分析

在这类问题中,信用评估机构要针对不同申请公司的各自特点提出信用评价,判断失误的例子经常发生,给信贷机构带来巨大损失。采用神经网络评价系统不仅评价结果具有较高的可信度,而且可以避免信贷分析人员的主观好恶和人情关系造成的错误。神经网络评价系统将公司贷款申请表中的关键数据编码为输入向量,将实际的信用情况作为输出评价,用数以千计的历史数据对网络进行训练后,可给出准确客观的评价结果。因此基于神经网络的评价系统在金融风险分析领域应用十分广泛。

(2) 市场预测

市场预测问题可归结为对影响市场供求关系的诸多因素的综合分析,以及对价格变化规律的掌握。应用神经网络进行市场预测的一类实例是期货市场的神经网络预测。其做法是根据某期货市场每月平均期货价格、价格不定性和市场心理指标量等因素,建立较为准确可靠的市场模型。该类模型不仅能判断价格的未来走势,而且能在走势持续一段时间后预测到价格的反转。神经网络市场预测在股票走势预测中也有广泛应用。

3.3 生物神经网络基础

人脑是物质世界进化的最高级产物,也是世界上最复杂的信息处理系统。其中,人脑生物神经系统是人脑实现各种高级功能的物质基础,也是人工神经网络力图模拟和借鉴的原型系统。为了研究开发具有某种人脑高级功能的仿脑智能系统,需要尽可能了解人脑神经系统的结构与机制,并从信息处理及工程学的观点进行分析、抽象与简化。

神经生理学和神经解剖学的研究表明,神经元是脑组织的基本单元,是神经系统结构与功能的单位。据估计,人类大脑大约包含有 1.4×10^{11} 个神经元,每个神经元与大约 $10^3 \sim 10^5$ 个其他神经元相连接,构成一个极为庞大而复杂的网络,即生物神经网络。生物神经网络中各神经元之间连接的强弱,按照外部的激励信号作自适应变化,而每个神经元又随着接收到的多个激励信号的综合结果呈现出兴奋与抑制状态。大脑的学习过程就是神经元之间连接强度随外部激励信息做自适应变化的过程,大脑处理信息的结果由各神经元状态的整体效果确定。显然,神经元是人脑信息处理系统的最小单元。

3.3.1 生物神经元的结构

人脑中神经元的形态不尽相同,功能也有差异,但从组成结构来看,各种神经元是有共性的。图 3-2 给出一个典型神经元的基本结构和与其他神经元发生连接的简化示意图。神经元在结构上由细胞体、树突、轴突和突触 4 部分组成。

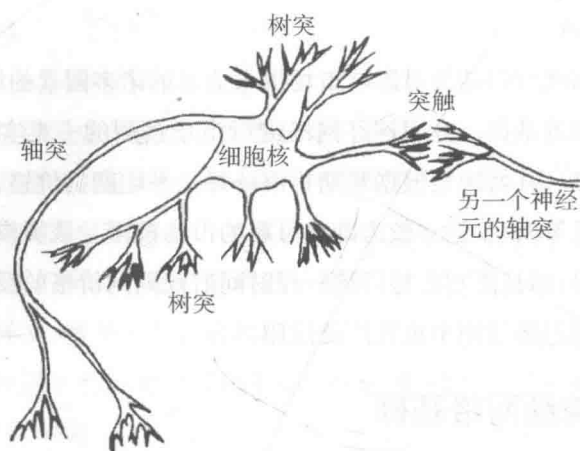


图 3-2 生物神经元简化示意图

(1) 细胞体(Cellbody)

细胞体是神经元的主体,由细胞核、细胞质和细胞膜 3 部分构成。细胞核占据细胞体的很大一部分,进行着呼吸和新陈代谢等许多生理过程。细胞体的外部是细胞膜,将膜内外细胞液分开。由于细胞膜对细胞液中的不同离子具有不同的通透性,使得膜内外存在着离子浓度差,从而出现内负外正的静息电位。

(2) 树突(Dendrite)

从细胞体向外延伸出许多突起的神经纤维,其中大部分突起较短,其分支多群集在细胞体附近形成灌木丛状,这些突起称为树突。神经元靠树突接受来自其他神经元的输入信号,相当于细胞体的输入端。

(3) 轴突(Axon)

由细胞体伸出的最长的一条突起称为轴突。轴突比树突长而细,用来传出细胞体产生的输出电化学信号。轴突也称神经纤维,其分支倾向于在神经纤维终端处长出,这些细的分支称为轴突末梢或神经末梢。每一条神经末梢可以向四面八方传出信号,相当于细胞体的输出端。

(4) 突触(Synapse)

神经元之间通过一个神经元的轴突末梢和其他神经元的细胞体或树突进行通信连接,这种连接相当于神经元之间的输入/输出接口,称为突触。突触包括突触前、突触间隙和突触后 3 个部分。突触前是第一个神经元的轴突末梢部

分,突触后是指第二个神经元的树突或细胞体等受体表面。突触在轴突末梢与其他神经元的受体表面相接触的地方有 $15\sim 50\text{nm}$ 的间隙,称为突触间隙,在电学上把两者断开,如图 3-2 所示。每个神经元大约有 $10^3\sim 10^5$ 个突触,多个神经元以突触连接,即形成神经网络。

3.3.2 生物神经元的信息处理机理

在生物神经元中,突触为输入输出接口,树突和细胞体为输入端,接受突触点的输入信号;细胞体相当于一个微型处理器,对各树突和细胞体各部位收到的来自其他神经元的输入信号进行组合,并在一定条件下触发,产生一输出信号;输出信号沿轴突传至末梢,轴突末梢作为输出端通过突触将这一输出信号传向其他神经元的树突和细胞体。下面对生物神经元之间接收、产生、传递和处理信息的机理进行分析。

1. 信息的产生

研究认为,神经元间信息的产生、传递和处理是一种电化学活动。由于细胞膜本身对不同离子具有不同的通透性,而造成膜内外细胞液中的离子存在浓度差。神经元在无神经信号输入时,其细胞膜内外因离子浓度差而造成的电位差为 -70mV (内负外正) 左右,称为静息电位,此时细胞膜的状态称为极化状态(Polarization),神经元的状态为静息状态。当神经元受到外界的刺激时,如果膜电位从静息电位向正偏移,称之为去极化(Depolarization),此时神经元的状态为兴奋状态;如果膜电位从静息电位向负偏移,称之为超级化(Hyperpolarization),此时神经元的状态为抑制状态。神经元细胞膜的去极化和超级化程度反映了神经元的兴奋和抑制的强烈程度。在某一给定时刻,神经元总是处于静息、兴奋和抑制 3 种状态之一。神经元中信息的产生与兴奋程度相关,在外界刺激下,当神经元的兴奋程度超过了某个限度,也就是细胞膜去极化程度超过了某个阈值电位时,神经元被激发而输出神经脉冲。每个神经脉冲产生的经过如下:当膜电位以静息膜电位为基准高出 15mV ,即超过阈值电位 (-55mV) 时,该神经细胞变成活性细胞,其膜电位自发地急速升高,在 1ms 内比静息膜电位上升 100mV 左右,此后膜电位又急速下降,回到静止时的值。这一过程称作细胞的兴奋过程,兴奋的结果产生一个宽度为 1ms ,振幅为 100mV

的电脉冲,又称神经冲动,如图 3-3 所示。

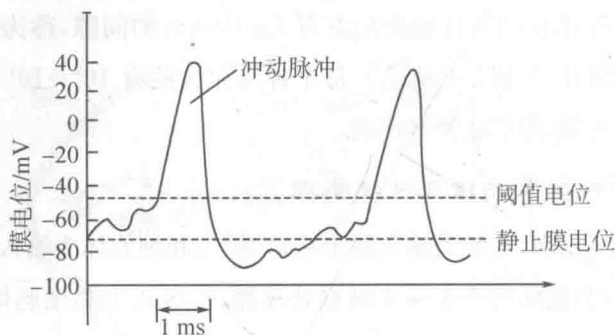


图 3-3 膜电位变化

值得注意的是,当细胞体产生一个电脉冲后,即使受到很强的刺激,也不会立刻产生兴奋,这是因为神经元发放电脉冲时阈值急速升高,持续 1ms 后慢慢下降到 -55mV 这一正常状态,这段时间约为数毫秒,称为不应期。不应期结束后,若细胞受到很强的刺激,则再次产生兴奋性电脉冲。由此可见,神经元产生的信息是具有电脉冲形式的神经冲动。各脉冲的宽度和幅度相同,而脉冲的间隔是随机变化的。某神经元的输入脉冲密度越大,其兴奋程度越高,在单位时间内产生的脉冲串的平均频率也越高。

2. 信息的传递与接收

神经脉冲信号沿轴突传向其末端的各个分支,在轴突的末端触及突触前时,突触前的突触小泡能释放一种化学物质,称为递质。在前一个神经元发放脉冲并传到其轴突末端后,这种递质从突触前膜释放出,经突触间隙的液体扩散,在突触后膜与特殊受体相结合。受体的性质决定了递质的作用是兴奋的还是抑制的,并据此改变后膜的离子通透性,从而使突触后膜电位发生变化。根据突触后膜电位的变化,可将突触分为两种:兴奋性突触和抑制性突触。兴奋性突触的后膜电位随递质与受体结合数量的增加而向正电位方向增大,抑制性突触的后膜电位随递质与受体结合数量的增加向更负电位方向变化。从化学角度看,当兴奋性化学递质传送到突触后膜时,后膜对离子通透性的改变使流入细胞膜内的正离子增加,从而使突触后成分去极化,产生兴奋性突触后电位;当抑制性化学递质传送到突触后膜时,后膜对离子通透性的改变使流出细胞膜外的正离子增加,从而使突触后成分超极化,产生抑制性突触后电位。

当突触前膜释放的兴奋性递质使突触后膜的去极化电位超过了某个阈电位时,后一个神经元就有神经脉冲输出,从而把前一神经元的信息传递给了后一神经元(如图 3-4 所示)。



图 3-4 突触信息传递过程

从脉冲信号到达突触前膜到突触后膜电位发生变化,有 $0.2\sim 1\text{ms}$ 的时间延迟,称为突触延迟(Synaptic Delay),这段延迟是化学递质分泌、向突触间隙扩散、到达突触后膜并在那里发生作用的时间总和。由此可见,突触对神经冲动的传递具有延时作用。

在人脑中,神经元间的突触联系大部分是在出生后由于给予刺激而成长起来的。外界刺激性质不同,能够改变神经元之间的突触联系,即突触后膜电位变化的方向与大小。从突触信息传递的角度看,表现为放大倍数和极性的变化。正是由于各神经元之间的突触连接强度和极性有所不同并可进行调整,因此人脑才具有学习和存储信息的功能。

3. 信息的整合

神经元对信息的接收和传递都是通过突触来进行的。单个神经元可以与多达上千个其他神经元的轴突末梢形成突触连接,接受从各个轴突传来的脉冲输入。这些输入可到达神经元的不同部位,输入部位不同,对神经元影响的权重也不同。在同一时刻产生的刺激所引起的膜电位变化,大致等于各单独刺激引起的膜电位变化的代数和。这种累加和称为空间整合。另外,各输入脉冲抵达神经元的先后时间也不一样。由一个脉冲引起的突触后膜电位很小,但在其持续时间内有另一脉冲相继到达时,总的突触后膜电位增大。这种现象称为时间整合。

输入一个神经元的信息在时间和空间上常呈现一种复杂多变的形式,神经元需要对它们进行积累和整合加工,从而决定其输出的时机和强弱。正是神经元的这种时空整合作用,才使得亿万个神经元在神经系统中可以有条不紊、夜

以继日地处理着各种复杂的信息,执行着生物中枢神经系统的各种信息处理功能。

4. 生物神经网络

由多个生物神经元以确定方式和拓扑结构相互连接即形成生物神经网络,它是一种更为灵巧、复杂的生物信息处理系统。研究表明,每一个生物神经网络系统均是一个有层次的、多单元的动态信息处理系统,它们有其独特的运行方式和控制机制,以接受生物系统内外环境的输入信息,加以综合分析处理,然后调节控制机体对环境做出适当反应。生物神经网络的功能不是单个神经元信息处理功能的简单叠加。每个神经元都有许多突触与其他神经元连接,任何一个单独的突触连接都不能完全表现一项信息。只有当它们集成总体时才能对刺激的特殊性质给出明确的答复。由于神经元之间突触连接方式和连接强度的不同并且具有可塑性,神经网络在宏观上呈现出千变万化的复杂的信息处理能力。

3.4 人工神经网络模型及其性能

3.4.1 人工神经元模型

人工神经网络是在现代神经生物学研究基础上提出的模拟生物过程,反映人脑某些特性的一种计算结构。它不是人脑神经系统的真实描写,而只是它的某种抽象、简化和模拟。根据前面对生物神经网络的介绍可知,神经元及其突触是神经网络的基本器件。因此,模拟生物神经网络应首先模拟生物神经元。在人工神经网络中,神经元常被称为“处理单元”。有时从网络的观点出发常把它称为“节点”。人工神经元是对生物神经元的一种形式化描述,它对生物神经元的信息处理过程进行抽象,并用数学语言予以描述;对生物神经元的结构和功能进行模拟,并用模型图予以表达。

1. 神经元的建模

目前人们提出的神经元模型已有很多,其中最早提出且影响最大的,是1943年心理学家 McCulloch 和数学家 W. Pitts 在分析总结神经元基本特性的基础上首先提出的 M-P 模型。该模型经过不断改进后,形成目前广泛应用的

形式神经元模型。关于神经元的信总处理机制,该模型在简化的基础上提出以下6点假定进行描述。

- (1)每个神经元都是一个多输入单输出的信息处理单元。
- (2)神经元输入分兴奋性输入和抑制性输入两种类型。
- (3)神经元具有空间整合特性和阈值特性。
- (4)神经元输入与输出间有固定的时滞,主要取决于突触延搁。
- (5)忽略时间整合作用和不应期。
- (6)神经元本身是非时变的,即其突触时延和突触强度均为常数。

显然,上述假定是对生物神经元信息处理过程的简化和概括,它清晰地描述了生物神经元信息处理的特点,而且便于进行形式化表达。上述假定,可用图3-5中的神经元模型示意图进行图解表示。

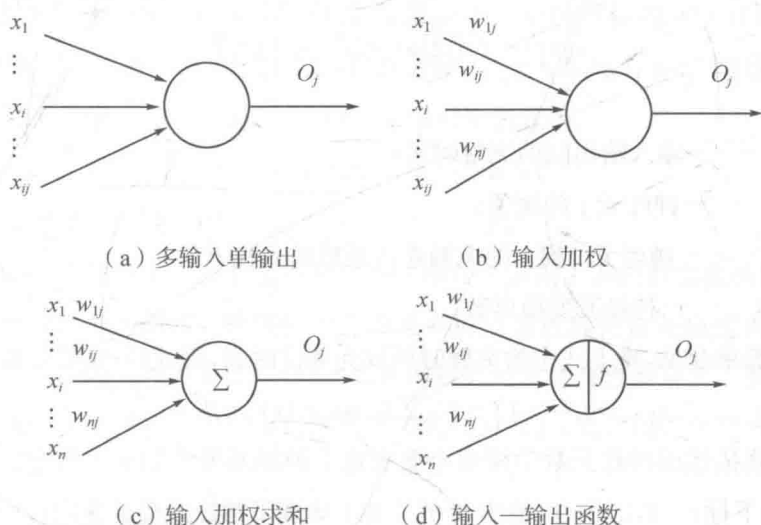


图3-5 神经元模型示意图

图3-5(a)表明,正如生物神经元有许多激励输入一样,人工神经元也应该有许多的输入信号(图中每个输入的大小用确定数值 x_i 表示),它们同时输入神经元 j 。生物神经元具有不同的突触性质和突触强度,其对输入的影响是使有些输入在神经元产生脉冲输出过程中所起的作用比另外一些输入更为重要。图3-5(b)中对神经元的每一个输入都有一个加权系数 w_{ij} ,称为权重值,其正负模拟了生物神经元中突触的兴奋和抑制,其大小则代表了突触的不同连接强

度。作为人工神经网络的基本处理单元,必须对全部输入信号进行整合,以确定各类输入的作用总效果,图 3-5(c)表示组合输入信号的“总和值”,对应于生物神经元的膜电位。神经元激活与否取决于某一阈值电平,即只有当其输入总和超过阈值时,神经元才被激活而发放脉冲,否则神经元不会产生输出信号。人工神经元的输出也同生物神经元一样仅有一个,如用 O_j 表示神经元输出,则输出与输入之间的对应关系可用图 3-5(d)中的某种函数来表示,这种函数一般都是非线性的。

2. 神经元的数学模型

上述内容可用一个数学表达式进行抽象与概括。令 $x_i(t)$ 表示 t 时刻神经元 j 接收的来自神经元 i 的输入信息, $o_j(t)$ 表示 t 时刻神经元 j 的输出信息,则神经元 j 的状态可表达为

$$o_j(t) = f\left\{\left[\sum_{i=1}^n w_{ij}x_i(t - \tau_{ij})\right] - T_j\right\} \quad (3-1)$$

式中,

τ_{ij} —— 输入输出间的突触时延;

T_j —— 神经元 j 的阈值;

w_{ij} —— 神经元 i 到 j 的突触连接系数或称权重值;

$f()$ —— 神经元变换函数。

为简单起见,将上式中的突触时延取为单位时间,则式(3-1)可写为

$$o_j(t+1) = f\left\{\left[\sum_{i=1}^n w_{ij}x_i(t)\right] - T_j\right\} \quad (3-2)$$

上式描述的神经元数学模型全面表达了神经元模型的 6 点假定。其中输入 x_i 的下标 $i=1,2,\dots,n$, 输出 o_j 的下标 j 体现了神经元模型假定(1)中的“多输入单输出”。权重值 w_{ij} 的正负体现了假定(2)中“突触的兴奋与抑制”。 T_j 代表假定(3)中神经元的“阈值”;“输入总和”常称为神经元在 t 时刻的净输入,用下式表示

$$net_j(t) = \sum_{i=1}^n w_{ij}x_i(t) \quad (3-3)$$

$net'_j(t)$ 体现了神经元 j 的空间整合特性而未考虑时间整合,当 $net'_j - T_j > 0$ 时,神经元才能被激活。 $O_j(t+1)$ 与 $x_i(t)$ 之间的单位时差代表所有神经元具有相同的、恒定的工作节律,对应于假定(4)中的“突触延搁”; w_{ij} 与时间无关

体现了假定(6)中神经元的“非时变”。

为方便起见,在后面用到式(3-3)时,常将其中的 (t) 省略。式(3-3)还可表示为权重向量 W_j 和输入向量 X 的点积

$$net_j(t) = w_j^T X \quad (3-4)$$

其中, W_j 和 X 均为列向量,定义为

$$W_j = (w_{1j}, w_{2j}, \dots, w_{nj})^T$$

$$X = (x_1, x_2, \dots, x_n)$$

如果令 $x_0 = -1, w_{0j} = T_j$, 则有 $-T_j = x_0 w_{0j}$, 因此净输入与阈值之差可表达为

$$net_j - T_j = net_j = \sum_{i=1}^n w_{ij} x_i = W_j^T X \quad (3-5)$$

显然,式(3-4)中列向量 W_j 和 X 的第一个分量的下标均从1开始,而式(3-5)中则从0开始。采用式(3-5)的约定后,净输入改写为 net_j , 与原来的区别是包含了阈值。综合以上各式,神经元模型可简化为

$$O_j = net_j = f(net_j) = f(W_j^T X) \quad (3-6)$$

3. 神经元的变换函数

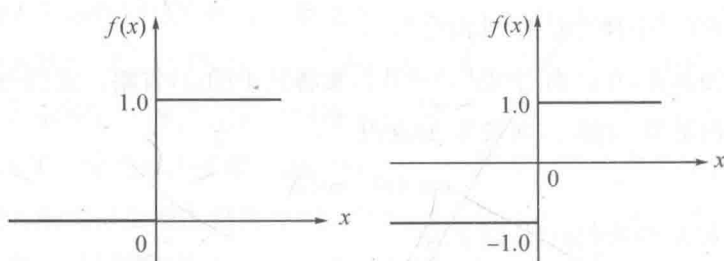
神经元的各种不同数学模型的主要区别在于采用了不同的变换函数,从而使神经元具有不同的信息处理特性。神经元的变换函数是决定人工神经网络整体性能的三大要素之一,因此变换函数的研究具有重要意义。神经元的变换函数反映了神经元输出与其激活状态之间的关系,最常用的变换函数有以下4种形式。

(1) 阈值型变换函数

阈值型变换函数采用了图3-6(a)中的单位阶跃函数,用下式定义:

$$f(t) = \begin{cases} 1; & x \geq 0 \\ 0; & x < 0 \end{cases} \quad (3-7)$$

具有这一作用方式的神经元称为阈值型神经元,这是神经元模型中最简单的一种,经典的M-P模型就属于这一类。函数中的自变量 x 代表 $net_j' - T_j$, 即当 $net_j' \geq T_j$ 时,神经元为兴奋状态,输出为1;当 $net_j' < T_j$ 时,神经元为抑制状态,输出为0。



(a)单极性阈值型变换函数 (b)双极性阈值型变换函数

图 3-6 阈值型变换函数

(2)非线性变换函数

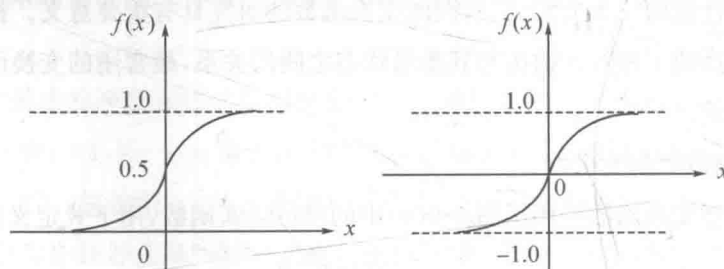
非线性变换函数为实数域 R 到 $[0, 1]$ 闭集的非减连续函数,代表了状态连续型神经元模型。最常用的非线性变换函数是单极性的 Sigmoid 函数曲线,简称 S 型函数,其特点是函数本身及其导数都是连续的,因而在处理上十分方便。单极性 S 型函数定义如下:

$$f(x) = \frac{1}{1+e^{-x}} \quad (3-8)$$

有时也常采用双极性 S 型函数(双曲正切)等形式

$$f(x) = \frac{2}{1+e^{-x}} - 1 = \frac{1-e^{-x}}{1+e^{-x}}$$

S 型函数其曲线特点见图 3-7。



(a)单极性S型变换函数 (b)双极性S型变换函数

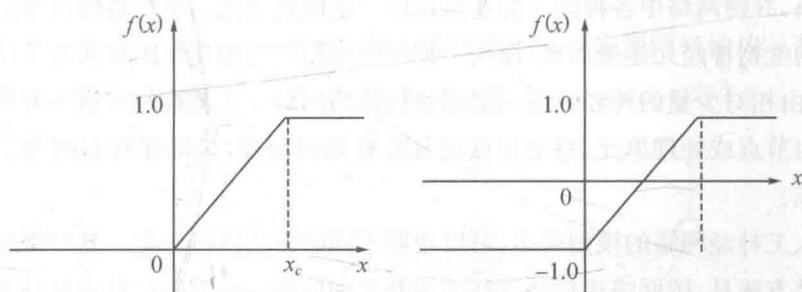
图 3-7 S 型变换函数

(3)分段线性变换函数

该函数的特点是神经元的输入与输出在一定区间内满足线性关系。由于具有分段线性的特点,因而在实现上比较简单。这类函数也称为伪线性函数,

单极性分段线性变换函数的表达式如下

$$f(x) = \begin{cases} 0 & x \leq 0 \\ cx & 0 < x \leq x_c \\ 1 & x_c < x \end{cases} \quad (3-9)$$



(a) 单极性分段性变换函数 (b) 双极性分段性变换函数

图 3-8 分段线性变换函数

(4) 概率型变换函数

采用概率型变换函数的神经元模型其输入与输出之间的关系是不确定的,需用一个随机函数来描述其输出状态为 1 或为 0 的概率。设神经元输出为 1 的概率为

$$P(1) = \frac{1}{1 + e^{-x/T}} \quad (3-10)$$

式中, T 为温度参数。由于采用该变换函数的神经元输出状态分布与热力学中的玻尔兹曼 (Boltzmann) 分布相类似, 因此这种神经元模型也称为热力学模型。

3.4.2 人工神经网络模型

神经细胞是构筑神经系统和人脑的基本单元, 它既具有结构和功能的动态特性, 又具有时间和空间的动态特性, 其简单有序的编排构成了完美复杂的大脑。神经细胞之间的通信是通过其具有可塑性的突触耦合实现的, 这使它们成为一个有机的整体。人工神经网络就是通过对人脑的基本单元——神经细胞——的建模和连接, 来探索模拟人脑神经系统功能的模型, 其任务是构造具有学习、联想、记忆和模式识别等智能信息处理功能的人工系统。

在各种智能信息处理模型中, 人工神经网络是最具有大脑风格的智能信息处理模型, 许多网络都能反映人脑功能的若干基本特性, 但并非生物系统的逼

真描述,只是对其局部电路的某种模仿、简化和抽象。

大量神经元组成庞大的神经网络,才能实现对复杂信息的处理与存储,并表现出各种优越的特性。神经网络的强大功能与其大规模并行互连、非线性处理以及互连结构的可塑性密切相关。因此必须按一定规则将神经元连接成神经网络,并使网络中各神经元的连接权按一定规则变化。生物神经网络由数以亿计的生物神经元连接而成,而人工神经网络限于物理实现困难和为了计算简便,是由相对少量的神经元按一定规律构成的网络。人工神经网络中的神经元常称为节点或处理单元,每个节点均具有相同的结构,其动作在时间和空间上均同步。

人工神经网络的模型很多,可以按照不同的方法进行分类。其中常见的两种分类方法是:按网络连接的拓扑结构分类和按网络内部的信息流向分类。

1. 网络拓扑结构类型

神经元之间的连接方式不同,网络的拓扑结构也不同。根据神经元之间连接方式,可将神经网络结构分为两大类。

(1) 层次型结构

具有层次型结构的神经网络将神经元按功能分成若干层,如输入层、中间层(也称为隐层)和输出层,各层顺序相连,如图 3-9 所示。输入层各神经元负责接收来自外界的输入信息,并传递给中间各隐层神经元;隐层是神经网络的内部信息处理层,负责信息变换,根据信息变换能力的需要,隐层可设计为一层或多层;最后一个隐层传递到输出层各神经元的信息经进一步处理后即完成一次信息处理,由输出层向外界(如执行机构或显示设备)输出信息处理结果。层次型网络结构有 3 种典型的结合方式。

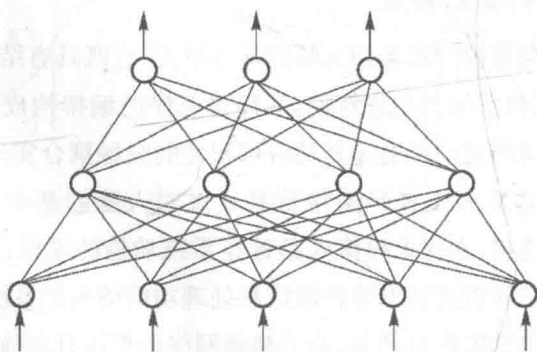


图 3-9 层次型网络结构示意图

①单纯型层次网络结构

在图 3-9 所示的层次型网络中,神经元分层排列,各层神经元接收前一层输入并输出到下一层,层内神经元自身以及神经元之间不存在连接通路。

②输出层到输入层有连接的层次网络结构

图 3-10 所示为输入层到输出层有连接路径的层次型网络结构。其中输入层神经元既可接收输入,也具有信息处理功能。

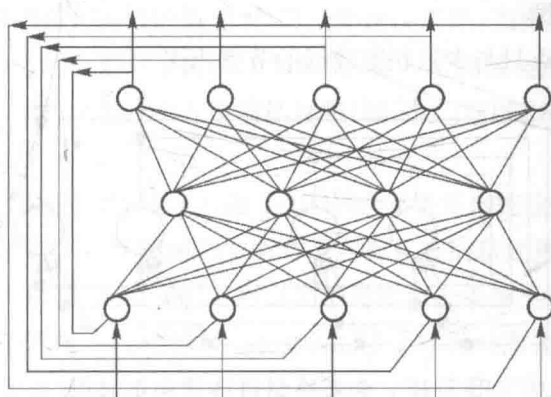


图 3-10 输出层到输入层有连接的层次型网络结构示意图

③层内有互连的层次网络结构

图 3-11 所示为同一层内神经元有互连的层次网络结构,这种结构的特点是在同一层内引入神经元间的侧向作用,使得能同时激活的神经元个数可控,以实现各层神经元的自组织。

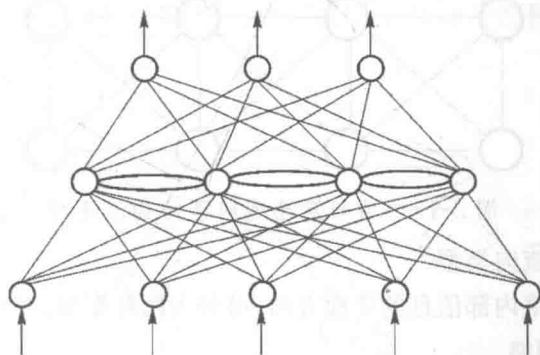


图 3-11 层内有连接的层次型网络结构示意图

(2)互连型结构

对于互连型网络结构,网络中任意两个节点之间都可能存在连接路径,因此可以根据网络中节点的互连程度将互连型网络结构细分为3种情况。

①全互连型

网络中的每个节点均与所有其他节点连接,如图3-12所示。

②局部互连型

网络中的每个节点只与其邻近的节点有连接,如图3-13所示。

③稀疏连接型

网络中的节点只与少数相距较远的节点相连。

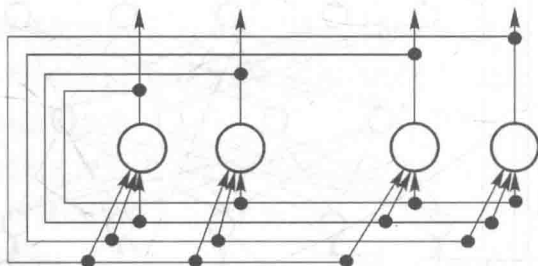


图 3-12 全互连型网络结构示意图

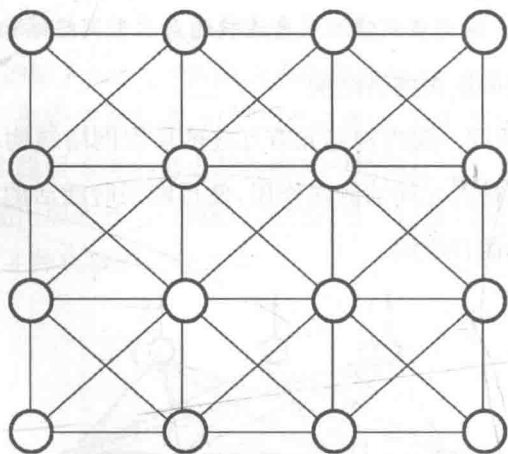


图 3-13 局部互连型网络结构示意图

2. 网络信息流向类型

根据神经网络内部信息的传递方向,可分为两种类型。

(1)前馈型网络

单纯前馈型网络的结构特点与图3-9中所示的分层网络完全相同,前馈是

因网络信息处理的方向是从输入层到各隐层再到输出层逐层进行而得名。从信息处理能力看,网络中的节点可分为两种:一种是输入节点,只负责从外界引入信息后向前传递给第一隐层;另一种是具有处理能力的节点,包括各隐层和输出层节点。前馈网络中某一层的输出是下一层的输入,信息的处理具有逐层传递进行的方向性,一般不存在反馈环路。因此这类网络很容易串联起来建立多层前馈网络。

多层前馈网络可用一个有向无环路的图表示。其中输入层常记为网络的第一层,第一个隐层记为网络的第二层,其余类推。所以,当提到具有单层计算神经元的网络时,指的应是一个两层前馈网络(输入层和输出层),当提到具有单隐层的网络时,指的应是一个三层前馈网络(输入层、隐层和输出层)。

(2) 反馈型网络

单纯反馈型网络的结构特点与图 3-10 中的网络结构完全相同,称为反馈网络是指其信息流向。在反馈网络中所有节点都具有信息处理功能,而且每个节点既可以从事外界接收输入,同时又可以向外界输出。单纯全互连结构网络是一种典型的反馈型网络,可以用图 3-14 所示的完全的无向图表示。

上面介绍的分类方法、结构形式和信息流向只是对目前常见的网络结构的概括和抽象。实际应用的神经网络可能同时兼有其中一种或几种形式。例如,从连接形式看,层次网络中可能出现局部的互连;从信息流向看,前馈网络中可能出现局部反馈。综合来看,图 3-9 至图 3-13 中的网络模型可分别称为:前馈层次型、输入输出有反馈的前馈层次型、前馈层内互连型、反馈全互连型和反馈局部互连型。

神经网络的拓扑结构是决定神经网络特性的第二大要素,其特点可归纳为分布式存储记忆与分布式信息处理、高度互连性、高度并行性和结构可塑性。

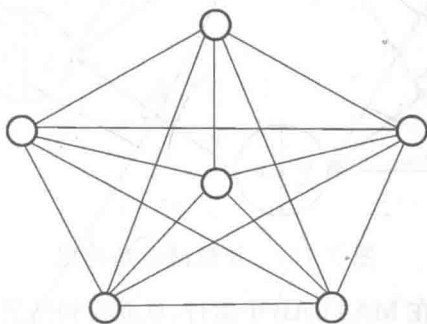


图 3-14 反馈型网络结构示意图

下面通过一个非线性分类问题来考察前馈型神经网络在模式识别领域的应用。

问题： x - y 平面上有 200 个点，分别属于两个类别。试设计并训练一个多层前向神经网络，以完成该分类任务，使得被错误分类的样本数量最低。 ω_1 类以 ‘+’ 标示， ω_2 类以 ‘○’ 标示，如图 3-15 所示。

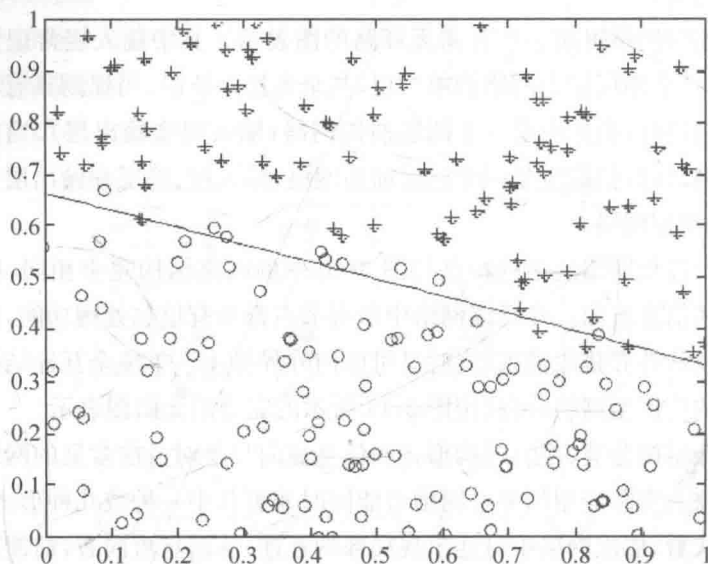


图 3-15 样本点在 x - y 空间示意图

神经网络结构设计如下：

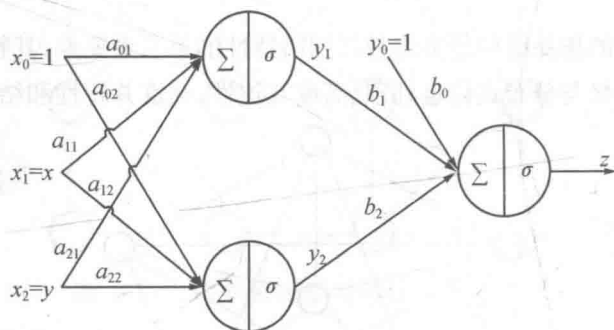


图 3-16 神经网络结构图

可参考以下程序在 MATLAB 中运行，从而得到结果：

```
clear x0 x1 x2;
```

```

beta=0.1;
miu=0;
[n m]=size(x);
for i=1:1:n
x0(1,i)=1;
x1(1,i)=x(i,1);
x2(1,i)=x(i,2);
y0(1,i)=1;
t(1,i)=x(i,3);
end
a01=rand();
a11=rand();
a21=rand();
a02=rand();
a12=rand();
a22=rand();
b0=rand();
b1=rand();
b2=rand();
delta_a01=0;
delta_a11=0;
delta_a21=0;
delta_a02=0;
delta_a12=0;
delta_a22=0;
delta_b0=0;
delta_b1=0;
delta_b2=0;
l=1;

```



```

while 1
    u1=a01 * x0+a11 * x1+a21 * x2;
    u2=a02 * x0+a12 * x1+a22 * x2;
    y1=2./(1+exp(-u1))-1;
    y2=2./(1+exp(-u2))-1;
    v=b0 * y0+b1 * y1+b2 * y2;
    z=2./(1+exp(-v))-1;
    error=0;
    for i=1:1:n
        if(z(1,i) > 0 && t(1,i)==1) || (z(1,i) < 0 && t(1,i)==-1)
            %0
        else error=error+1;
        end
    end
    error
    temp0 = -(t-z). * exp(-v)./(1+exp(-v)).^2;
    temp1 = b1. * exp(-u1)./(1+exp(-u1)).^2;
    temp2 = b2. * exp(-u2)./(1+exp(-u2)).^2;
    delta_b0=miu * delta_b0+(1-miu) * beta * sum(-temp0. * y0);
    delta_b1=miu * delta_b1+(1-miu) * beta * sum(-temp0. * y1);
    delta_b2=miu * delta_b2+(1-miu) * beta * sum(-temp0. * y2);
    delta_a01 = miu * delta_a01 + (1-miu) * beta * sum(-temp0. *
temp1. * x0);
    delta_a11 = miu * delta_a11 + (1-miu) * beta * sum(-temp0. *
temp1. * x1);
    delta_a21 = miu * delta_a21 + (1-miu) * beta * sum(-temp0. *
temp1. * x2);
    delta_a02 = miu * delta_a02 + (1-miu) * beta * sum(-temp0. *
temp2. * x0);

```

```
delta_a12 = miu * delta_a12 + (1 - miu) * beta * sum(-temp0. *  
temp2. * x1);
```

```
delta_a22 = miu * delta_a22 + (1 - miu) * beta * sum(-temp0. *  
temp2. * x2);
```

```
b0 = b0 + delta_b0;
```

```
b1 = b1 + delta_b1;
```

```
b2 = b2 + delta_b2;
```

```
a01 = a01 + delta_a01;
```

```
a11 = a11 + delta_a11;
```

```
a21 = a21 + delta_a21;
```

```
a02 = a02 + delta_a02;
```

```
a12 = a12 + delta_a12;
```

```
a22 = a22 + delta_a22;
```

```
l = l + 1;
```

```
if l == 1000
```

```
break;
```

```
end
```

```
end
```

```
j1 = 1;
```

```
j2 = 1;
```

```
k1 = 1;
```

```
k2 = 1;
```

```
for i = 1:1:n
```

```
if x(i,3) == -1
```

```
if z(1,i) < 0
```

```
x11(j1,:) = x(i,:);
```

```
j1 = j1 + 1;
```

```
else
```

```
x12(j2,:) = x(i,:);
```

```
j2=j2+1;
end
else
if z(1,i) > 0
x21(k1,:)=x(i,:);
k1=k1+1;
else
x22(k2,:)=x(i,:);
k2=k2+1;
end
end
end
hold on;
plot(x11(:,1),x11(:,2),'g*');
plot(x12(:,1),x12(:,2),'r+');
plot(x21(:,1),x21(:,2),'bo');
plot(x22(:,1),x22(:,2),'r+');
hold off;
axis([0 1 0 1]);
```

程序运行结果,经过训练,该神经网络对 200 个样本的分类正确率达到了 96.5%,分类效果较好,具体分类情况如下图所示,其中被错误分类的样本已用“×”标示出,其他正确分类的样本仍用原类别对应的符号进行标示,如图 3-16 所示。

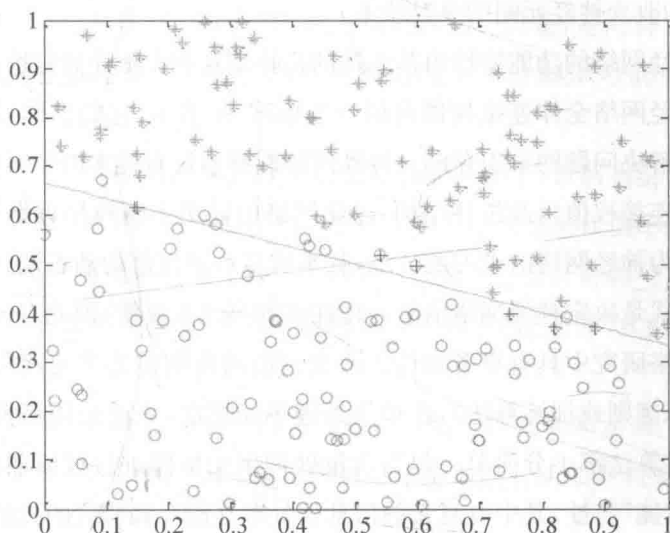


图 3-16 程序运行结果示意图

3.5 神经网络学习

人类具有学习能力,人的知识和智慧是在不断的学习与实践逐渐形成和发展起来的。关于学习,可定义为:根据与环境的相互作用而发生的行为改变,其结果导致对外界刺激产生反应的新模式的建立。

学习过程离不开训练,学习过程就是一种经过训练而使个体在行为上产生较为持久改变的过程。例如,游泳等体育技能的学习需要反复的训练才能提高,数学等理论知识的掌握需要通过大量的习题进行练习。一般来说,学习效果随着训练量的增加而提高,这就是通过学习获得的进步。

关于学习的神经机制,涉及神经元如何分布、处理和存储信息。这样的问题单用行为研究是不能回答的,必须把研究深入到细胞和分子水平。正如两位心理学家 D. O. Hebb 和 J. Konorski 提出的,学习和记忆一定包含有神经回路的变化。每一种心理功能(如记忆与思想),均归因于神经细胞组群的活动。在大脑中,要建立功能性的神经元连接,突触的形成是关键。神经元之间的突触联系,其基本部分是先天就有的,但其他部分是由于学习过程中频繁地给予刺激而形成的。突触的形成、稳定与修饰均与刺激有关,外界给予的刺激性质不

同,能形成和改变神经元间的突触联系。

人工神经网络的功能特性由其连接的拓扑结构和突触连接强度,即连接权值决定。神经网络全体连接权值可用一个矩阵 W 表示,它的整体反映了神经网络对于所解决问题的知识存储。神经网络能够通过对样本的学习训练,不断改变网络的连接权值以及拓扑结构,以使网络的输出不断地接近期望的输出。这一过程称为神经网络的学习或训练,其本质是可变权值的动态调整。神经网络的学习方式是决定神经网络信息处理性能的第三大要素,因此有关学习的研究在神经网络研究中具有重要地位。改变权值的规则称为学习规则或学习算法(亦称训练规则或训练算法),在单个处理单元层次,无论采用哪种学习规则进行调整,其算法都十分简单。但当大量处理单元集体进行权值调整时,网络就呈现出“智能”特性,其中有意义的信息就分布存储在调节后的权值矩阵中。

神经网络的学习算法很多,根据一种广泛采用的分类方法,可将神经网络的学习算法归纳为 3 类:一类是有导师学习,一类为无导师学习,还有一类是灌输式学习。

有导师学习也称为有监督学习,这种学习模式采用的是纠错规则。在学习训练过程中需要不断地给网络成对提供一个输入模式和一个期望网络正确输出的模式,称为“教师信号”。将神经网络的实际输出同期望输出进行比较,当网络的输出与期望的教师信号不符时,根据差错的方向和大小按一定的规则调整权值,以使下一步网络的输出更接近期望结果。对于有导师学习,网络在能执行工作任务之前先经过学习,当网络对于各种给定的输入均能产生所期望的输出时,即认为网络已经在导师的训练下“学会”了训练数据集中包含的知识和规则,可以用来进行工作了。

无导师学习也称为无监督学习,学习过程中,需要不断地给网络提供动态输入信息,网络能根据特有的内部结构和学习规则,在输入信息流中发现任何可能存在的模式和规律,同时能根据网络的功能和输入信息调整权值,这个过程称为网络的自组织,其结果是使网络能对属于同一类的模式进行自动分类。在这种学习模式中,网络的权值调整不取决于外来教师信号的影响,可以认为,网络的学习评价标准隐含于网络的内部。

在有导师学习中,提供给神经网络学习的外部指导信息越多,神经网络学

会并掌握的知识越多,解决问题的能力也就越强。但是,有时神经网络所解决的问题的先验信息很少,甚至没有,这种情况下无导师学习就显得更有实际意义。

灌输式学习是指将网络设计成能记忆特别的例子,以后当给定有关该例子的输入信息时,例子便被回忆起来。灌输式学习中网络的权值不是通过训练逐渐形成的,而是通过某种设计方法得到的。权值一旦设计好即一次性“灌输”给神经网络不再变动,因此网络对权值的“学习”是“死记硬背”式的,而不是训练式的。

有导师学习和无导师学习网络的运行一般分为训练阶段和工作阶段两个阶段。训练学习的目的是为了从训练数据中提取隐含的知识和规律,并存储于网络中供工作阶段使用。

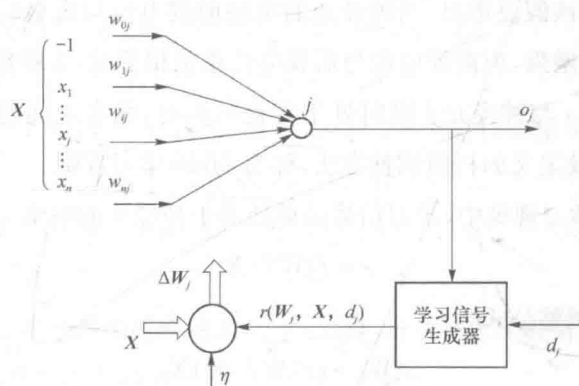


图 3-17 权值调整的一般情况

可以认为,一个神经元是一个自适应单元,其权值可以根据它所接收的输入信号、它的输出信号以及对应的监督信号进行调整。日本著名神经网络学者 Amari 于 1990 年提出一种神经网络权值调整的通用学习规则,该规则的图解表示如图 3-17 所示。图中的神经元 j 是神经网络中的某个节点,其输入用向量 X 表示,该输入可以来自网络外部,也可以来自其他神经元的输出。第 i 个输入与神经元 j 的连接权值用 w_{ij} 表示,连接到神经元 j 的全部权值构成了权向量 W_j 。应当注意的是,该神经元的阈值 $T_j = w_{0j}$,对应的输入分量 x_0 恒为 -1 。图中, $r = r(W_j, X, d_j)$ 代表学习信号,该信号通常是 W_j 和 X 的函数,而在有导师学习时,它也是教师信号 d_j 的函数。通用学习规则可表达为:权向量

W_j 的在 t 时刻的调整量 $\Delta W_j(t)$ 与 t 时刻的输入向量 $X(t)$ 和学习信号 r 的乘积成正比。用数学式表示为。

$$\Delta W_j = \eta r [W_j(t), X(t), d_j(t)] X(t) \quad (3-11)$$

式中, η 为正数, 称为学习常数, 其值决定了学习速率。基于离散时间调整时, 下一时刻的权向量应为

$$W_j(t+1) = W_j(t) + \eta r [W_j(t), X(t), d_j(t)] X(t) \quad (3-12)$$

不同的学习规则对 $r(W_j, X, d_j)$ 有不同的定义, 从而形成各种各样的神经网络。下面对常用学习算法做简要介绍, 其具体应用将在后续各章中展开。

3.5.1 Hebb 学习规则

1949 年, 心理学家 D. O. Hebb 最早提出了关于神经网络学习机理的“突触修正”的假设。该假设指出, 当神经元的突触前膜电位与突触后膜电位同时为正时, 突触传导增强, 当前膜电位与后膜电位正负相反时, 突触传导减弱。也就是说, 当神经元 i 与神经元 j 同时处于兴奋状态时, 两者之间的连接强度应增强。根据该假设定义的权值调整方法, 称为 Hebb 学习规则。

在 Hebb 学习规则中, 学习信号简单地等于神经元的输出

$$r = f(WT_j X) \quad (3-13)$$

权向量的调整公式为

$$\Delta W_j = \eta f(WT_j X) X \quad (3-14a)$$

权向量中, 每个分量的调整由下式确定

$$\Delta w_{ij} = \eta f(WT_j X) x_i = \eta o_j x_{ii} = 0, 1, \dots, n \quad (3-14b)$$

上式表明, 权值调整量与输入输出的乘积成正比。显然, 经常出现的输入模式将对权向量有最大的影响。在这种情况下, Hebb 学习规则需预先设置权饱和值, 以防止输入和输出正负始终一致时出现权值无约束增长。

此外, 要求权值初始化, 即在学习开始前 ($t=0$), 先对 $W_j(0)$ 赋予零附近的小随机数。

Hebb 学习规则代表一种纯前馈、无导师学习。该规则至今仍在各种神经网络模型中起着重要作用。

下面用一个简单的例子说明 Hebb 学习规则的应用。

例 3.1 设有 4 输入单输出神经网络,其阈值 $T=0$,学习率 $\eta=1$,3 个输入样本向量和初始权向量分别为 $X_1=(1,-2,1.5,0)T$, $X_2=(1,-0.5,-2,-1.5)T$, $X_3=(0,1,-1,1.5)T$, $W(0)=(1,-1,0,0.5)T$ 。

解首先设变换函数为双极性离散函数 $f(net) = \text{sgn}(net)$, 权值调整步骤为:

(1) 输入第一个样本 X_1 , 计算净输入 net_1 , 并调整权向量 $W(1)$:

$$net_1 = W(0)TX_1 = (1, -1, 0, 0.5)(1, -2, 1.5, 0)T = 3$$

$$W(1) = W(0) + \eta \text{sgn}(net_1)X_1 = (1, -1, 0, 0.5)T + (1, -2, 1.5, 0)T = (2, -3, 1.5, 0.5)T$$

(2) 输入第二个样本 X_2 , 计算净输入 net_2 , 并调整权向量 $W(2)$:

$$net_2 = W(1)TX_2 = (2, -3, 1.5, 0.5)(1, -0.5, -2, -1.5)T = -0.25$$

$$W(2) = W(1) + \eta \text{sgn}(net_2)X_2 = (2, -3, 1.5, 0.5)T - (1, -0.5, -2, -1.5)T = (1, -2.5, 3.5, 2)T$$

(3) 输入第三个样本 X_3 , 计算净输入 net_3 , 并调整权向量 $W(3)$:

$$net_3 = W(2)TX_3 = (1, -2.5, 3.5, 2)(0, 1, -1, 1.5)T = -3$$

$$W(3) = W(2) + \eta \text{sgn}(net_3)X_3 = (1, -2.5, 3.5, 2)T - (0, 1, -1, 1.5)T = (1, -3.5, 4.5, 0.5)T$$

可见,当变换函数为符号函数且 $\eta=1$ 时,Hebb 学习规则的权值调整将简化为权向量加或减输入向量。

下面设变换函数为双极性连续函数 $f(net) = (1 - e^{-net}) / (1 + e^{-net})$, 权值调整步骤同上。

$$(1) \quad net_1 = W(0)TX_1 = 3$$

$$o_1 = f(net_1) = (1 - e^{-net}) / (1 + e^{-net}) = 0.905$$

$$W(1) = W(0) + \eta f(net_1)X_1 = (1, -1, 0, 0.5)T + (0.905)(1, -2, 1.5, 0)T$$

$$(2) \quad net_2 = W(1)TX_2 = -0.154$$

$$o_2 = f(net_2) = (1 - e^{-net}) / (1 + e^{-net}) = -0.077$$

$$W(2) = W(1) + \eta f(net_2)X_2 = (1, -1, 0, 0.5)T + (-0.077)(1, -0.5, -2, -1.5)T$$

$$(3) \quad net_3 = W(2)TX_3 = -3.36$$

$$o_3 = f(net_3) = (1 - e^{-net}) / (1 + e^{-net}) = -0.932$$

$$W(3)=W(2)+\eta f(\text{net}3)X_3=(1.828,-3.70,2.44,-0.785)T$$

比较两种权值调整结果可以看出,两种变换函数下的权值调整方向是一致的,但采用连续变换函数时,权值调整力度减弱。

3.5.2 离散感知器学习规则

1958年,美国学者 Frank Rosenblatt 首次定义了一个具有单层计算单元的神经网络结构,称为感知器(Perceptron)。感知器的学习规则规定,学习信号等于神经元期望输出(教师信号)与实际输出之差

$$r=d_j-o_j \quad (3-15)$$

式中, d_j 为期望的输出, $o_j=f(W_j^T X)$ 。感知器采用了与阈值变换函数类似的符号变换函数,其表达为

$$f(W_j^T \hat{\circ} X)=\{\text{mathop{\rm sgn}\,}\}(W_j^T \hat{\circ} X)=\begin{cases} 1 & W_j^T \hat{\circ} X \geq 0 \\ -1 & W_j^T \hat{\circ} X < 0 \end{cases} \quad (3-16)$$

因此,权值调整公式应为

$$\Delta W_j=\eta[d_j-\text{sgn}(W_j^T X)]X \quad (3-17a)$$

$$\Delta W_{ij}=\eta[d_j-\{\text{sgn}(W_j^T \hat{\circ} X)\}x_i] \quad i=0,1,\dots,n \quad (3-17b)$$

式中,当实际输出与期望值相同时,权值不需要调整;在有误差存在情况下,由于 d_j 和 $\text{sgn}(W_j^T X) \in \{-1,1\}$,权值调整公式可简化为

$$\Delta W_j=\pm 2\eta X \quad (3-17c)$$

感知器学习规则只适用于二进制神经元,初始权值可取任意值。

感知器学习规则代表一种有导师学习。由于感知器理论是研究其他神经网络的基础,该规则对于神经网络的有导师学习具有极为重要的意义。

3.5.3 连续感知器学习规则

1986年,认知心理学家 McClelland 和 Rumelhart 在神经网络训练中引入了 δ 规则,该规则亦可称为连续感知器学习规则,与上述离散感知器学习规则并行。 δ 规则的学习信号规定为

$$r=[d_j-f(W_j^T X)]f'(W_j^T X)=(d_j-o_j)f'(\text{net}_j) \quad (3-18)$$

上式定义的学习信号称为 δ 。式中, $f'(W_j^T X)$ 是变换函数 $f(\text{net}_j)$ 的导

数。显然, δ 规则要求变换函数可导, 因此只适用于有导师学习中定义的连续变换函数, 如 Sigmoid 函数。

事实上, δ 规则很容易由输出值与期望值的最小平方误差条件推导出来。定义神经元输出与期望输出之间的平方误差为

$$E = 1/2(d_j - o_j)^2 = 1/2[d_j - f(W_j TX)]^2 \quad (3-19)$$

其中, 误差 E 是权向量 W_j 的函数。欲使误差 E 最小, W_j 应与误差的负梯度成正比, 即

$$\Delta W_j = -\eta \Delta E \quad (3-20)$$

式中, 比例系数 η 是一个正常数。由式 3-19, 误差梯度为

$$\Delta E = -(d_j - o_j) f'(W_j TX) X \quad (3-21)$$

将此结果代入式(3-20), 可得权值调整计算式

$$\Delta W_j = \eta(d_j - o_j) f'(net_j) X \quad (3-22a)$$

可以看出, 上式中 η 与 X 之间的部分正是式(3-18)中定义的学习信号 δ 。 ΔW_j 中每个分量的调整由下式计算

$$\Delta w_{ij} = \eta(d_j - o_j) f'(net_j) x_i, i = 0, 1, \dots, n \quad (3-22b)$$

δ 学习规则可推广到多层前馈网络中, 权值可初始化为任意值。

下面举例说明 δ 学习规则的应用。

例 3.2 设有 3 输入单输出神经元网络, 将阈值含于权向量内, 故有 $w_0 = T, x_0 = -1$, 学习率 $\eta = 0.1$, 3 个输入向量和初始权向量分别为 $X_1 = (-1, 1, -2, 0)T, X_2 = (-1, 0, 1.5, -0.5)T, X_3 = (-1, 1, 0.5, -1)T, W(0) = (0.5, 1, -1, 0)T$

解: 设变换函数为双极性连续函数 $f(net) = (1 - e^{-net}) / (1 + e^{-net})$, 权值调整步骤为

(1) 输入样本 X_1 , 计算净输入 net_1 及权向量 $W(1)$

$$net_1 = W(0)TX_1 = 2.5$$

$$o_1 = f(net_1) = (1 - e^{-net}) / (1 + e^{-net}) = 0.848$$

$$f'(net_1) = 1/2(1 - (o_1)^2) = 0.14$$

$$W(1) = W(0) + \eta(d_1 - o_1) f'(net_1) X_1 = (0.526, 0.974, -0.948, 0)T$$

(2) 输入样本 X_2 , 计算净输入 net_2 及权向量 $W(2)$

$$net_2 = W(1)TX_2 = -1.948$$

$$o_2 = f(net_2) = (1 - e^{-net}) / (1 + e^{-net}) = -0.75$$

$$f'(net_2) = 1/2(1 - (o_2)^2) = 0.218$$

$$W(2) = W(1) + \eta(d_2 - o_2)f'(net_2)X_2 = (0.531, 0.974, -0.956, 0.002, 0.531)^T$$

(3) 输入样本 X_3 , 计算净输入 net_3 及权向量 $W(3)$

$$net_3 = W(2)TX_3 = -2.416$$

$$o_3 = f(net_3) = (1 - e^{-net}) / (1 + e^{-net}) = -0.842$$

$$f'(net_3) = 1/2(1 - (o_3)^2) = 0.145$$

$$W(3) = W(2) + \eta(d_3 - o_3)f'(net_3)X_3 = (0.505, 0.947, -0.929, 0.016)^T$$

3.5.4 最小均方学习规则

1962年, Bernard Widrow 和 Marcian Hoff 提出了 Widrow 唱 Hoff 学习规则。因为它能使神经元实际输出与期望输出之间的平方差最小, 所以又称为最小均方规则(LMS)。LMS 学习规则的学习信号为

$$r = d_j - W_jTX \quad (3-23)$$

权向量调整量为

$$\Delta W_j = \eta(d_j - W_jTX)X \quad (3-24a)$$

ΔW_j 的各分量为

$$\Delta w_{ij} = \eta(d_j - W_jTX)x_j \quad i=0, 1, \dots, n \quad (3-24b)$$

实际上, 如果在 δ 学习规则中假定神经元变换函数为 $f(W_jTX) = W_jTX$, 则有

$$f'(W_jTX) = 1$$

此时式(3-18)与式(3-23)相同。因此, LMS 学习规则可以看成是 δ 学习规则的一个特殊情况。该学习规则与神经元采用的变换函数无关, 因而不需要对变换函数求导数, 不仅学习速度较快, 而且具有较高的精度。权值可初始化为任意值。

3.5.5 相关学习规则

相关学习规则规定学习信号为

$$r = d_j \quad (3-25)$$

易得出 ΔW_j 及 Δw_{ij} 分别为

$$\Delta W_j = \eta d_j X \quad (3-26a)$$

$$\Delta w_{ij} = \eta d_j x_i \quad i=0,1,\dots,n \quad (3-26b)$$

该规则表明,当 d_j 是 x_i 的期望输出时,相应的权值增量 Δw_{ij} 与两者的乘积 $d_j x_i$ 成正比。

如果 Hebb 学习规则中的变换函数为二进制函数,且有 $o_j = d_j$,则相关学习规则可看作 Hebb 规则的一种特殊情况。应当注意的是,Hebb 学习规则是无导师学习,而相关学习规则是有导师学习。这种学习规则要求将权值初始化为零。

3.5.6 胜者为王学习规则

胜者为王(Winner-Take-All)学习规则是一种竞争学习规则,用于无导师学习。一般将网络的某一层确定为竞争层,对于一个特定的输入 X ,竞争层的所有 p 个神经元均有输出响应,其中响应值最大的神经元 j 倡为在竞争中获胜的神经元,即

$$W_m TX = i = m_1, 2a, x?, p(W_m TX) \quad (3-27)$$

只有获胜神经元才有权调整其权向量 W_j 倡,调整量为

$$\Delta W_m = \alpha (X - W_m) \quad (3-28)$$

式中, $\alpha \in (0, 1]$, 是一个小的学习常数,一般其值随着学习的进展而减小。由于两个向量的点积越大,表明两者越近似,所以调整获胜神经元权值的结果是使 W_m 进一步接近当前输入 X 。显然,当下次出现与 X 相像的输入模式时,上次获胜的神经元更容易获胜。在反复的竞争学习过程中,竞争层的各神经元所对应的权向量被逐渐调整为输入样本空间的聚类中心。

在有些应用中,以获胜神经元为中心定义一个获胜邻域,除获胜神经元调整权值外,邻域内的其他神经元也不同程度地调整权值。权值一般被初始化为任意值并进行归一化处理。

3.5.7 外星学习规则

神经网络中有两类常见节点,分别称为内星节点和外星节点,其特点见图 3-18。图 3-18(a)中的内星节点总是接受来自其他神经元的输入加权信号,因此是信号的汇聚点,对应的权值向量称为内星权向量;图 3-18(b)中的外星节点总是向其他神经元发出输出加权信号,因此是信号的发散点,对应的权值向量称为外星权向量。内星学习规则规定内星节点的输出响应是输入向量 X 和内星权向量 W_j 的点积。该点积反映了 X 与 W_j 的相似程度,其权值按式 (3-28) 调整。因此 Winner-Take-All 学习规则与内星规则一致。下面介绍外星学习规则。

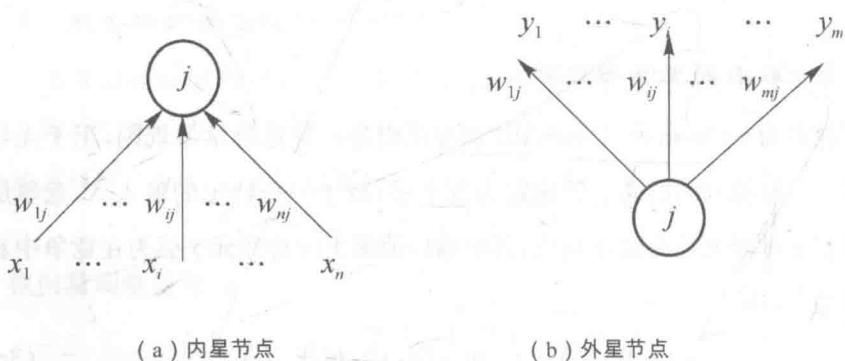


图 3-18 内星节点与外星节点

外星学习规则属于有导师学习,其目的是为了生成一个期望的 m 维输出向量 d , 设对应的外星权向量用 W_j 表示,学习规则如下

$$\Delta W_j = \eta(d - W_j) \quad (3-29)$$

式中, η 的规定与作用和式 (3-28) 中的 α 相同。正像式 (3-28) 给出的内星学习规则使节点 j 对应的内星权向量向输入向量 X 靠拢一样, 式 (3-29) 给出的外星学习规则使节点 j 对应的外星权向量向期望输出向量 d 靠拢。

以上集中介绍了神经网络中几种常用的学习规则,有些规则之间有着内在联系,读者可通过比较体会其异同。对上述各种学习规则的对比总结列于表 3-1 中。

表 3-1 常用学习规则一览表

学习规则	权值调整		权值初始化	学习方式	变换函数
	向量式	元素式			
Hebbian	$\Delta W_j = \eta f(W_j^T X) X$	$\Delta W_{ij} = \eta f(W_j^T X) X_i$	0	无导师	任意
离散 Perceptron	$\Delta W_j = \eta [d_j - \text{sgn}(W_j^T X)] X$	$\Delta W_{ij} = \eta [d_j - \text{sgn}(W_j^T X)] x_i$	任意	有导师	二进制
连续感知 器 δ 规则	$\Delta W_j = \eta (d_j - o_j) f(\text{net}_j) X$	$\Delta W_{ij} = \eta (d_j - o_j) f(\text{net}_j) x_i$	任意	有导师	连续
最小均 方 LMS	$\Delta W_j = \eta (d_j - W_j^T X) X$	$\Delta W_{ij} = \eta (d_j - W_j^T X) x_i$	任意	有导师	任意
相关 Correlation	$\Delta W_j = \eta d_j X$	$\Delta W_{ij} = \eta d_j x_i$	0	有导师	任意
胜者为王 Winner-Take-All	$\Delta W_j * = \eta (X - W_j *)$	$\Delta W_{ij} * = \eta (x_j - W_{ij} *)$	随机、 归一化	无导师	连续
外星 Outstar	$\Delta W_j = \eta (d - W_j)$	$\Delta W_{kj} = \eta (d_k - W_{kj})$	0	有导师	连续

3.6 神经网络系统设计与软硬件实现

人工神经网络的发展为探索新型智能计算机开辟了一条崭新的途径,要使神经网络的并行处理能力得以充分发挥,还有赖于其硬件实现。然而,目前各类神经计算机的研究还远未成熟,应用神经网络解决实际问题时主要靠软件进行虚拟实现。因此了解并掌握神经网络软件设计与开发方面的基本知识对于提高应用神经网络解决问题的能力水平和水平至关重要。本章主要从系统设计和软件开发与运行等几个方面加以阐述。

3.6.1 神经网络系统总体设计

一个设计良好的神经网络系统能代表问题求解的系统方法。开发神经网络系统的总体设计过程中应考虑这样几个问题:① 首先分析哪类问题需要使用神经网络;② 神经网络系统的整体处理过程的设计,即系统总图;③ 系统需求分析;④ 设计系统的各项性能指标;⑤ 预处理问题。下面就这几个问题进行讨论。

1. 神经网络的适用范围

当确定一个问题要用神经网络解决后,接着就要确定用什么样的网络模型和算法。如果有一组确知分类的输入模式数据,就可通过训练 BP 网开始试探解决问题。若不知道答案(分类)应该是什么,可从某种自组织学习网络结构入手。试验时可尝试使用不同的网络结构和网络参数,如学习率或动量系数等,并对其效果进行比较。

神经网络在应用中常常作为一个子系统在系统中的一个或多个位置出现,系统中的一个或多个神经网络往往起着各种各样的作用,在系统的详细设计过程中要尽可能开放思路,考虑不同的作用与组合。事实上,在许多应用中都使用若干个网络或多次使用网络,还有可能采用子网络构造大结构,甚至不同的网络也可拓扑组合成一个单一的结构。例如,用自组织网络对数据进行预处理,然后用其输出节点作为执行最终分类的反向传播网络的输入节点。又如,神经网络可作为专家系统中的数据预处理子系统,或作为从原始数据中提取参数的特征提取子系统。有时需要将多个网络模型结合使用,其中每个网络均作为综合网中的子网出现。总之神经网络在实际应用中存在许多可能的形式,因此应用神经网络解决问题时要打开思路。

2. 神经网络的设计过程与需求分析

神经网络的设计开发过程可以用图 3-19 所示的系统总图来描述。设计过程要完成的工作任务有 3 项:首先,是系统需求分析;其次,是数据准备,包括训练与测试数据的选择、数据特征化和预处理以及产生模式文件,在此过程中强调要求系统的最终用户参加,目的是保证训练数据和测试结果的有效性;再次是与计算机有关的任务,包括软件编程与系统调试等内容。

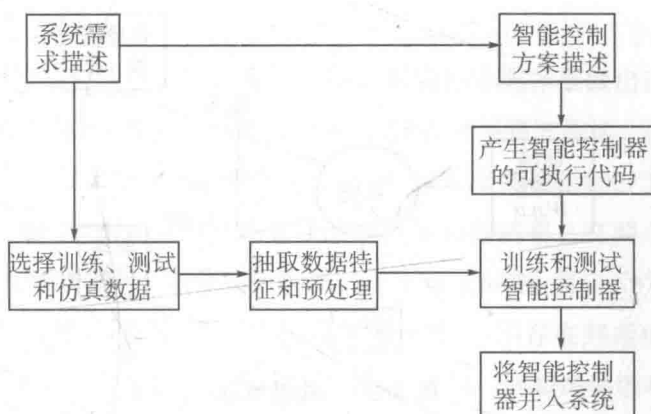


图 3-19 神经网络系统开发总图

系统需求分析一般应包括以下内容。

(1) 系统需求说明。系统分析是系统开发过程中最重要的工作之一，因为此阶段的错误和疏忽会对项目产生代价昂贵的后果。系统分析阶段的产物是系统详细需求分析文档，以便准确描述系统的行为和评估完成状况。

(2) 结构化分析。结构化分析使用一套工具来产生结构化需求说明，由数据流图、数据词典和结构化文字几部分组成。数据流图显示的是在系统和环境间以及处理过程间的信息和控制信号流，并将需求模型图形化和生动化。使用结构化文字强调的是可读性而不是自动分析的能力，其目的是在某种程度上能与不懂计算机的用户沟通。

(3) 层次结构。数据流图是分等级按层次构造的，可用一些相互关联的图表示。如图 3-20 将整个系统看成单一的处理过程以及系统与环境间的数据流，图 3-21 将单一过程扩展，揭示了关于系统模型的更详尽的细节，包括一些过程、流和数据存储。图 3-21 中的每个过程都能够依次扩展成更详细的图表，分解可一直继续至任意的细节水平直至最终使用结构化文字能够充分描述的最低层次。层次结构是系统建模和系统构造的有力工具，它证明了结构化分析是较有效的系统描述技术。

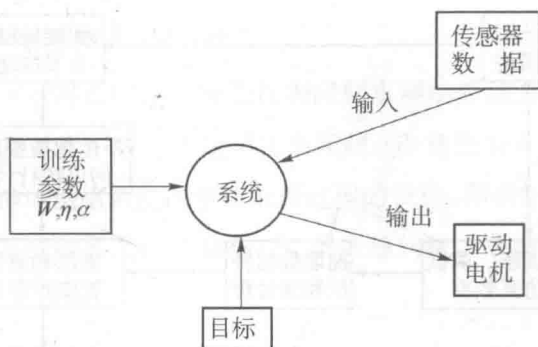


图 3-20 数据流图

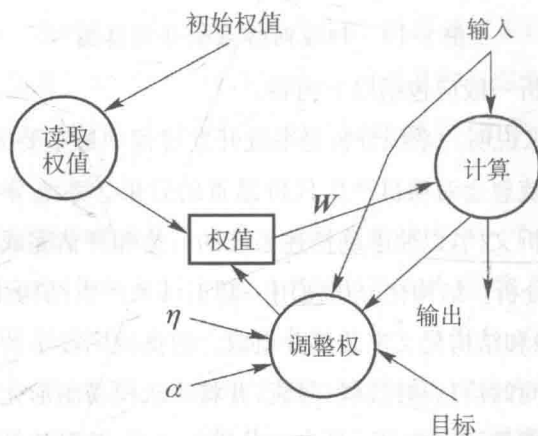


图 3-21 处理过程的细化

(4) 数据词典。数据词典是结构化分析的主要工具,目前普遍被用作一种称为系统的百科全书的软件工程数据库,以存储数据元素说明以及系统模型中对象与方法的需求说明。

3. 神经网络的性能评价

为了评价一个系统的运行质量,需要把对系统进行测试运行时得到的数据和已建立的标准相比较。为了研究有关神经网络的运行质量,必须首先建立一些能反映其质量的性能指标,这些指标应对不同的网络具有通用性和可比性。目前在这方面尚缺乏系统而深入的研究,但仍可借鉴相关领域的运行检测技术。下面简要介绍关于神经网络性能评价的几个常用指标,而应用时选用哪一种指标取决于系统的类型以及使用者的技术水平等因素。

(1) 百分比正确率

神经网络运行的百分比正确率就是根据某种分类标准做出正确判断的百分比。神经网络用于模式识别和分类等问题时,常用到该指标。但在某些神经网络应用中,百分比正确率的概念不太适用,应采用其他指标。为了计算正确率应选择合适的分类标准和有代表性的训练集和测试集。有两个因素会影响以上选择的合理性:一个是分类标准本身的不确定问题,另一个是样本集的代表性。例如,当神经网络用于对印刷体字母分类时,不存在判断标准的不确定性问题。但是有些分类任务会存在较大的主观因素,例如烤烟烟叶质量定级的分类,随专家的观点不同分类结果会略有差异。在用神经网络检测癫痫棘波时,6位神经科医生中任何2位共同认定的单个棘波的平均一致率仅为60%。因此在分类之前统一观点是十分重要的,这需要系统最终用户的积极参与,才能正确建立统一的分类标准。训练集和测试集样本的代表性是目前神经网络开发工作正在研究的课题之一。训练集和测试集的样本及由专家认定的代表类必须分布在所有类的范围内,包括那些在判断临界点附近的样本。设计者的人为因素也是非常重要的,应该避免设计者闭门造车地进行代表样本的分类确定工作,而要让系统的用户参与这一过程。虽然设计人员能为用户提供系统运行的技术要求等信息,但在样本设计和整个设计过程中都应该让用户尽可能发挥作用。在测试和训练中要使用不同的样本集,当样本数不充足时,可以循环利用已有的样本进行训练和测试。此外,所选的训练集应该使每种样本的分类结果具有相同的数目。即,如果神经网络有3个输出节点,对于每一次分类有一个相应的节点激活,则训练样本集中每种分类结果的样本数目应该定为总样本数目的三分之一。

(2) 均方误差

神经网络的均方误差为总误差除以样本总数,而总误差定义为

$$E = \frac{1}{2} \sum_{j=1}^m \sum_{p=1}^p (d_j^p - o_j^p)^2 \quad (3-30)$$

应用反向传播算法训练神经网络的目的是使均方误差最小。在应用均方误差时,应注意两种情况:第一,均方误差的定义公式中包括乘积因子1/2,但是在许多应用场合都省略了该因子,因此在比较各种不同的神经网络时应注意

均方误差的计算中是否包含乘积因子 $1/2$; 第二, 误差项对所有输出节点求和时会产生一个潜在的问题, 均方误差无法精确地反映具有不同节点数的神经网络结构之间的差别。如果训练一个单输出节点的神经网络能达到一固定误差, 而训练一个结构基本相同的多输出节点的神经网络时, 误差可能会增大, 这是因为均方误差定义为除以训练集或测试集中的样本数而不是除以节点数。在某些应用场合, 用户要求计算每个节点的误差, 可以定义节点平均均方误差为 (样本平均) 均方误差除以输出节点数。由于平均节点均方误差主要用于反向传播算法, 所以它主要用于 BP 网络的性能评价。

(3) 归一化误差

Pinda 提出了一种与神经网络结构无关, 取值为 $0 \sim 1$ 的误差标准 E_{mean} 。定义为

$$E_{\text{mean}} = \frac{1}{2} \sum_{j=1}^m \sum_{p=1}^P (d_j^p - \bar{d}_j)^2 \quad (3-31)$$

式中, $\bar{d}_j$ 为所有样本在第 j 个输出节点的期望输出值的平均值, d_j^p 是第 j 个输出节点的期望输出值, P 为样本总数, m 为输出节点数。则归一化误差 En 定义为式 (3-30) 的总误差 E 除以上式的 E_{mean}

$$En = E / E_{\text{mean}} \quad (3-32)$$

归一化误差对 BP 神经网络十分有用。当神经网络“猜测”正确的输出值是平均目标值时, 出现“最坏的情况”是 $En = 1$ 。当样本学习结束后, En 的值趋向于零, 其速度取决于神经网络的结构。归一化误差反映的是基于误差的输出方差的比例, 而与神经网络本身的结构 (包括初始化的随机权值) 无关。因此, 在大多数场合, 归一化误差标准是 BP 神经网络中最有价值的误差标准之一。

(4) 接收操作特性曲线

评价神经网络系统的另一个途径是接收操作特性曲线 (ROC)。ROC 曲线用来反映系统某一个输出节点在做出一个判断时的正确性, 因此下面的讨论集中于单输出节点网络。若用判断的阳性和阴性表示将某一输入样本判断为某类的肯定与否定, 一个给定输出神经元所表示的判断存在 4 种可能性, 列于表 3-2 中。第一种可能性称为真阳性判断 (TP), 即系统的阳性判断与根据标准得到的阳性判断相一致, 比如系统鉴别出神经科医生确认的癫痫棘波; 第二种可

能性称为假阳性判断(FP),即系统做出阳性判断而标准做出阴性判断,例如系统判断出的癫痫棘波神经科医生认为不对;第三种可能性是假阴性判断(FN),即标准做出阳性判断而系统作出阴性的判断,例如神经科医生鉴别出的癫痫棘波系统却未找出;第四种可能性是真阴性判断(TN),即系统和标准都做出阴性判断,如系统和神经科医生都判断不存在癫痫棘波。

表 3-2 ROC 曲线定义中的可能性表

	标准判断		
	阳性	阴性	
阳性	TP	FP	
阴性	FN	TN	

利用上述这 4 种可能性的两种比例可绘出 ROC 曲线。第一种比例是 TP/(TP+FN),称为真阳性率(在某些应用场合称为灵敏度);第二种比例是 FP/(FP+TN),称为假阳性率。ROC 曲线由真阳性率轴和假阳性率轴上的点连接而成。为了画出真阳性率/假阳性率坐标轴中的点,可对输出节点设置不同的判断阈值。对于每个选定的阈值,统计出系统判断结果的真阳性率和假阳性率作为 ROC 曲线上点的坐标值。图 3-22 给出两种不同结构的神经网络的 ROC 曲线,曲线 NNT2 代表的系统比 NNT1 所代表的系统整体运行性能更好。坐标轴对角线上的虚线表示真/假阳性率相等,即无法判断的情况。

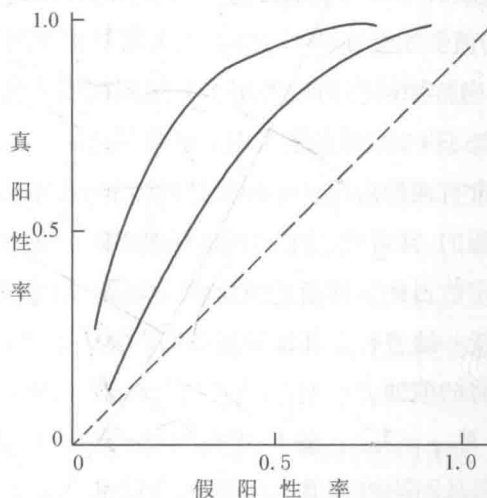


图 3-22 ROC 曲线示例

如果用单一指标评价系统运行情况,可以通过计算 ROC 曲线下所包围的面积决定,这实际上是用 ROC 曲线来评价系统运行性能的主要方法。整图的面积是一个单位方格,ROC 曲线以下的面积是整图的一个部分,曲线以下的面积必定在 0.5~1.0 之间,前者是当系统无法判断时对角线以下部分的面积,后者是当系统判断完全正确时曲线以下的面积。一种简单的计算方法是用直线线段连接相邻的点,并计算梯形折线以下的面积。为得到较光滑的 ROC 曲线,大约需要 9~10 个点。

(5) 灵敏度、精度和特异度

灵敏度是指实际存在的事物能被检测到的可能性,也称为回忆度,其定义与 ROC 曲线定义中的真阳性率相同。在某些要求防止出现漏检事件的场合,如在预后严重的疾病检测中,该指标变得非常重要。精度是系统所做的正确的阳性判断数目除以系统做出的所有阳性判断的总数,在表 3-2 中,就是 $TP/(TP+FP)$,它包含着假阳性判断的强度。特异度是指一件实际不存在的事物被检测为不存在的可能性,定义为 $TN/(FP+TN)$,或称为真阴性率。可以看出,灵敏度、精度和特异度是表 3-2 中 4 种数量的另一种表达方式。

4. 输入数据的预处理

在设计神经网络时,预处理是最难处理的问题之一。一方面预处理有许多种类;另一方面预处理有许多种实现方法。大多数神经网络通常需要归一化的输入,即每个输入的值要始终在 0~1 之间,或者每个输入向量的长度要为常量(如 1);前者用于反向传播网络而后者用于自组织网络。虽然对 BP 网的输入归一化的必要性看法不一,但对多数应用来说归一化是一种好的做法。

数据归一化通常有两种情况。一种常见的情况是,输入 BP 网各个输入节点的原始数据是同源的,常常代表了时间间隔的取样。例如,电压波形以一定的频率采样得到一定数目的采样值成组地输入网络。在这种情况下,归一化必须在所有的通道上统一地进行。如果数据分布在最大值($X_{\max}$)与最小值($X_{\min}$)之间,则首先把所有的值加上 $-X_{\min}$,使它们分布于 0 和 $(X_{\max} - X_{\min})$ 之间,然后再用每个值除以 $X_{\max} - X_{\min}$ 。这样所有的值被归一化成 0~1 之间的数。 $X_{\max}$ 和 $X_{\min}$ 很可能是从不同的通道上获得的,所以又称为交叉通道归一化。第二种常见的情况是,用不同种类参数作为输入,如电压、持续时间、波形的尖

峰参数等。更有可能是一些统计参数如标准方差、相关系数和 K 方检验参数等。这种情况下,跨所有通道的归一化会使网络的训练失败。例如,某些通道表示波形尖锐度,它们只能从 $-0.1 \sim +0.1$ 变化。其他通道表示波形振幅,能从 $-50 \sim +50$ 之间变化,显然归一化后尖锐度将被振幅所淹没。每种类型通道中若存在 0.1 个单位的偏差,归一化后将变为 0.001 单位的偏差。0.1% 的动态范围在振幅通道里可能很容易接受,然而在波形尖锐度中,0.1 的偏差代表了 50% 的动态变化,所以会严重影响网络训练或测试时对尖锐度参数的分辨能力。当输入为不同种类的参数时,对每个通道单独归一化的优点是每个通道可在 $0 \sim 1$ 区间反映其动态变化范围,缺点是任意两个通道之间的关系偏离了一个偏移量和多重因素的范围,因此每个通道单独归一化有时会在训练网络时造成困难。例如,用从生物电位波形中提取的参数训练网络,其中两个参数是振幅,3 个是宽度,3 个是尖锐度,另一个参数是斜率。振幅的测量单位是伏特,宽度是秒,而尖锐度是度数。对于宽度参数,把其中两个相加成为过零点间半波的宽度,第三个是半波二阶导数的两点间宽度。在原始数据中,前两个宽度之和总是大于第三个宽度,而其中任意一个又小于第三个宽度。这一类关系以及在两个振幅或 3 个尖锐度参数之间存在的类似的其他关系,在单独通道的归一化处理中就被遮掩了。

3.6.2 神经网络的软件实现

神经网络编程语言既可用高级语言也可用低级语言。C 语言是神经网络应用软件的基本编程工具,但其他高级语言也同样适用。汇编语言虽然只在程序代码中占很小一部分,但它是开发人员的另一种基本工具。它常用于提高神经网络的已有功能或解决与硬件相关的难点。

运行神经网络软件时主要涉及以下几个问题。

(1) 网络的输入—输出。神经网络一般采用文本文件保存输入—输出模式对和权值,其优点在于用户能直接阅读和用字处理工具进行编辑,而且能通过磁盘互相传递,通过打印机输出或通过电子邮件进行传递;另一个优点是容易输入数据库并进行分析。文本文件的缺点是占用字节数多,每个数据可能占用多达 10 个字符,而用浮点二进制数表示仅需 4 个字节。

(2)数据归一化。输入模式必须先归一化后再输入神经网络,较好的方式是把归一化处理放在预处理阶段进行,而不在神经网络内做归一化,这样不仅能减少神经网络的代码,而且也具有灵活方便、可移植性和通用性强的优点。

(3)连接权。输入网络的连接权和阈值是一组初值。输入循环的形式是

$for(i=0; i \leq nInputNode; i++)$

C语言中数组的 N 个元素按 $0 \sim N-1$ 标记,对于各隐层和输出层的每个神经元,上述循环从相应的输入层读取每个连接权和阈值,阈值存储在权向量的第一项或最后一项。开始训练时,权值随机分布在一个很小的范围之内(如 ± 0.3),权初始值文件可以由一个随机数产生器产生。训练结束后,把最后得到的权值矩阵写入输出文件。输出文件和输入样本文件相仿,通常是文本文件。以BP网为例,把权值写入文件有两个原因:第一,训练结束后获得的权值代表了系统学习的结果,设计者可以利用这些权值来测试系统识别和分类新模式的能力。通常,网络对一组训练模式训练一次,然后反复运用训练结果来对新模式进行识别。在识别过程中网络进行的是正向传递运算,对每一个模式仅需迭代运算一次。在训练时,网络必须进行正向和反向运算,反复迭代导致运算量急剧增大,因此可以在大型机上训练大型神经网络,然后把训练得到的权值文件送到PC机上运行神经网络。这是一种十分有价值的机器之间传递“学习”的技术,一台机器通过训练所得的结果可以快速传送到另一台机器,在不同的环境下运行神经网络。第二,训练结束后保存下来的权值可以重新输入网络继续训练。在某些情况下需要重新训练权值,对它进行修正以改善神经网络的性能,这时只需将以前保存的权值作为网络的初始权值进行训练。

(4)特性参数。神经网络的特性参数包括网络的拓扑结构、处理单元数、转移函数类型、学习规则、学习率和动量因子等。其中多数特性参数设计是由代码实现的算法,后在运行过程中难以更改。但学习率和动量因子等参数可以在运行过程中进行修改。

(5)运行状况监测。反向传播算法的目的是使均方误差最小,一种较好的观察收敛趋势的方法是用显示出误差相对于迭代次数的曲线。通常用均方误差作为最基本的网络运行监测目标,但有时其他监测指标也十分有价值,例如神经网络的净输入和权值。神经网络的状态可以在运行时连续显示,即在每一

次迭代后刷新。由于显示占用了神经网络的处理时间,所以在实用系统中不太理想。另外,如果网络的迭代速度太快时,无法看清屏幕上快速变化的内容。通常,可以设定经过许多次迭代后刷新一次显示状态,或者暂停运行来观察“冻结”的显示状态。另外一种方法就是在神经网络运行结束后非实时地显示。

3.6.3 神经网络的高级开发环境

许多研究人员在开发自己的神经网络软件时都会碰到这样的问题:花在人工编码和跟踪调试上的巨大工作量大大影响了真正要进行的模型设计工作。开发环境能快捷方便地建立一个新的网络模型,通过试验立即获得反馈值,从而可以评价所设计的模型的运行结果。神经网络开发环境是设计调试网络模型的非常有用的工具,可以大大缩短开发时间,提高工作效率。

目前神经网络的开发方法有以下3种模式。

(1)人工编码。神经网络开发的一种常用模式是人工编码,即研究人员针对某个具体问题选择一种算法,然后由人工编制代码在计算机上实现一个神经网络系统。这种开发方式在前面介绍过。

(2)算法库。神经网络开发的第二种模式是使用神经网络算法库。神经网络开发工具有各种有效的算法可供研究人员选择,同时神经网络的实现是自动形成的。因此,研究人员很容易实现所设计的神经网络。选择算法实际上是从算法库中提取了一个代码的结构,然后填上代码段(Fragment)以规定网络的特性参数。

3)生成网络模型。神经网络开发的第三种模式是生成神经网络模型,使得研究人员能建立任何设想的网络而不仅是在库中定义的算法。

1. 神经网络的开发环境及其特征

理想的神经网络开发环境应具有使用简单、功能强大、有效性和可扩展性等关键特征。因此开发环境应具有描述和运行网络模型的良好用户界面,使研究人员不必掌握操作系统或实现神经网络模型的计算机硬件知识就能进行网络模型的开发。开发环境应允许研究人员选择网络模型及其特性或定义新的网络模型及其特性,应能执行、监视、显示和控制神经网络的运行,并能将网络与其他处理功能连接。有效性是指神经网络开发环境要尽可能有效地使用

计算机。可扩展性意味着能定义和建立新网络类型的网络原始结构。

(1) 网络设计类

solvelin 线性网络设计

solver 径向基网络设计

solverbe 精确的径向基网络设计

solvehophopfield 网络设计

(2) 变换函数类

hardlim 硬限幅传递函数

hardlims 对称硬限幅传递函数

purelin 线性传递函数

tansig 正切 s 型传递函数

logsig 对数 s 型传递函数

satlin 饱和线性传递函数

satlins 对称饱和线性传递函数

radbas 径向基传递函数

dist 计算向量间的距离

compet 自组织映射传递函数

dpurelin 线性传递函数的导数

dtansig 正切 s 型传递函数的导数

dlogsig 对数 s 型传递函数的导数

(3) 学习规则类

learnp 感知层学习规则

learnpn 规范感知层学习规则

learnbp bp 学习规则

learnbpm 带动量项的 bp 学习规则

learnlm levenberg-marquardt 学习规则

learnwh widrow-hoff 学习规则

learnk kohonen 学习规则

learncon conscience 阈值学习函数

learnsom 自组织映射权学习函数

learnh hebb 学习规则

learnhd 退化的 hebb 学习规则

learnis 内星学习规则

learnos 外星学习规则

(4) 初始化类

initp 感知层初始化

lnitff 前向网络初始化

initlin 线性层初始化

initism 自组织映射网络的初始化

lmtelm elman 递归网络的初始化

lnitlay 层与层之间的网络初始化函数

mltwb 阈值与权值的初始化函数

lmtzero 零树阈值的初始化函数

lnltnw nguyen-widrow 层的初始化函数

lmtcon conscience 阈值的初始化函数

nudpoint 中点权值初始化函数

nwlog 对具有对数 s 型函数的节点产生 nguyen-widrow 随机数

nwtan 对具有正切 s 型函数的节点产生 nguyen-widrow 随机数

randnc 产生归一化列随机数

randnr 产生归一化行随机数

rands 产生对称随机数

(5) δ 函数类

deltalin 对 purelin 神经元的 δ 函数

deltatan 对 ansitg 神经元的 δ 函数

deltalog 对 logsig 神经元的 δ 函数

getdelta 对给定函数的 δ 函数

(6) 网络输入函数

netsum 网络输入函数的求和

dnetsum 网络输入求和函数导数

(7)性能分析函数

mae 均值绝对误差性能分析函数

mse 均方差性能分析函数

msereg 均方差 w/reg 性能分析函数

dmse 均方差性能分析函数的导数

msereg 均方差 w/reg 性能分析函数的导数

(8)网络训练类

trainwb 网络权与阈值的训练函数

traingd 梯度下降的 bp 算法训练函数

traingdm 梯度下降 w/动量的 bp 算法训练函数

traingda 梯度下降 w/自适应 lr 的 bp 算法训练函数

taingdx 梯度下降 w/动量和自适应 lr 的 bp 算法训练函数

trainlm levenberg-marquardt 的 bp 算法训练函数

trainwbl 每个训练周期用一个权值向量或偏差向量的训练函数

trainc 训练竞争层

trainfm 训练特性图

trainlvq 训练 lvq 网络

~nbpx 利用快速反向传播训练网络

trainelm 训练 elman 递归网络

trainsm 训练自组织映射网络

trainp 利用感知规则训练感知层

trainpn 利用规范感知规则训练感知层

trainbp 用 bp 算法训练前向网络

trainbpx 用快速 bp 算法训练前向网络

trainelm 训练 elman 递归网络

trainlm 用 levenberg-marquardt 算法训练前向网络

trainwh 用 widrow-hoff 规则训练线性层

(9)分析类

errsurf 计算误差曲面

maxlinlr 计算 widrow 唱 hoff 准则训练线性网络的最大学习率

presflop 计算浮点运算的表达式

(10)邻域函数类

nbdist 使用向量距离的邻域阵

nbpid 使用栅格距离的邻域阵

nbman 使用 manhattan 距离的邻域阵

neighb 1d 一维邻域函数

neighb 2d 二维邻域函数

(11)符号变换函数

ind2vec 转换下标成为向量

vec2ind 转换向量成为下标向量

(12)矩阵类

normc 计算归一化列矩阵

normr 计算归一化行矩阵

pnormc 计算伪归一化列矩阵

sumsq 计算平方和

quant 离散化成某数值的整数倍

delaysig 从信号矩阵中建立退化的信号矩阵

combvec 创建所有的向量集

(13)绘图类

plotpv 绘制具有 i/o 目标的输入向量图

plotpc 在已存在的感知器分类图上划上分类线

plotes 绘制误差曲面

plotep 在误差曲面上绘制出权值和阈值的位置

plotsom 绘制自组织映射图

barer 每个输出向量的误差条形图表

hintonw 绘制权值图

hintonwb 绘制权值和偏差图

ploterr 绘出网络误差与时间的关系
plotfa 绘出目标模式及网络函数的逼近
plotlr 绘出网络学习速率与训练次数的关系
plotmap 绘出具有任意邻近函数的特性
plottr 绘出网络误差记录及自适应学习速率
plotv 绘出始于坐标原点的单位模长向量线
plotvec 用不同颜色绘制向量
plotsm 绘制自组织映射图

(14) 仿真类

slmuc 竞争层仿真
slmuelm elman 递归网络仿真
simuff 前向网络仿真
simuhop hopfield 网络仿真
simulin 线性层仿真
slump 感知层仿真
simurb 径向基网络仿真
slmusm 自组织映射仿真

(15) 网络创建函数

newp 创建感知器网络
newlind 设计一个线性层
newlin 创建一个线性层
newff 创建一个前馈 bp 网络
newcf 创建一个多层前馈 bp 网络
newfftd 创建一个前馈输入延迟 bp 网络
newrb 设计一个径向基网络
nwerbe 设计一个严格的径向基网络
newgmnn 设计一个广义回归神经网络
newpnn 设计一个概率神经网络
newc 创建一个竞争层

newsom 创建一个自组织特征映射

newhop 创建一个 hopfield 递归网络

newelm 创建一个 elman 递归网络

(16)网络应用函数

sim 仿真一个神经网络

init 初始化一个神经网络

adapt 神经网络的自适应化

train 训练一个神经网络

(17)拓扑函数

gridtop 网格层拓扑函数

hextop 六角层拓扑函数

randtop 随机层拓扑函数

(18)自适应函数

adaptwb 网络权与阈值的自适应函数

adaptwh 用 widrow-hoff 规则自适应调节线性层的权

(19)权函数

dotprod 权函数的点积

ddotprod 权函数点积的导数

dist euclidean 距离权函数

normprod 规范点积权函数

negdist negative 距离权函数

mandist manhattan 距离权函数

linkdist link 距离权函数

3. 其他神经网络开发环境简介

下面简要介绍其他已有的神经网络开发环境。

(1)Plexi 神经网络开发环境

Plexi 是一个内置 Lisp 机的功能强大的综合系统,不能运行于个人计算机。该系统具有许多有价值的优点,既能支持高级用户试验新模型,又能满足初学者使用已定义的神经网络的要求。它有 2 个高分辨率的显示器显示图形

用户界面,图形网络编辑器能通过层图标上击键和通过神经束连接各层来表达结构。网络的作用可通过几个图形工具(如 Hinton 图、图形和点图)来观察。学习规则、转移函数和更新规则等特性可以通过描述获得,但同时也提供缺省值。该软件包括一些标准神经网络模型,但也可以自行定义神经网络模型。只要会使用 Lisp 语言,就能设计任何神经网络。

(2) Neuroshell 神经网络开发环境

Neuroshell 是一个有价值而又使用简便的神经网络应用工具,它具有内置反向传播模型的软件包,可以快速地开发应用程序,并有详细样例供研究、演示和实验。Neuro-shell 具有图形界面,可以对若干天的连续运行数据进行绘图和显示,用以分析趋势和因素之间的关系。

(3) Neuralworks Professional II 神经网络开发环境

Neuralworks Professional II 是一个昂贵的综合软件包,具有图形用户界面,容易生成新的神经网络或从大量已定义的网络类型中选择所需要的神经网络,可以方便地描述层的数量和节点数量等参数。图形显示界面上能显示网络的每个神经元节点和连接,网络结构也可在图形屏幕上进行编辑。可以用标准二进制码、ASCII 文件或用户自行编写的 C 函数预处理数据训练或测试网络,能用条形图和直方图显示网络运行结果。“探针”功能可用来观察网络内部的连接权、隐含的激励和 Hinton 图等。

(4) NETSET II 神经网络开发环境

该软件包运行于个人计算机的 Windows 环境下,用屏幕上从左到右排列的、用箭头连接的小图标代表对象。NETSET II 能良好地表示系统级的数据流程图,可以通过图标建立系统模型,图标有数据库、管道、动态数据交换、数据变换、神经网络和图形显示设备等。对象之间用反映信息流方向的箭头表示,绘制好系统流程图后,必须定义处理细节,如数据文件的输出、数据类型、预定义的格式等。描述一个神经网络对象可以从 19 个已封装的神经网络“实现”中选择。该系统有一个称为“Style”的功能,它可在任何级别由用户自行定义任何对象作为样本,命名并存储后可重复使用。

(5) N-NET210 神经网络开发环境

N-NET210 系统只有有指导学习和无指导学习两种固定的特征算法。开

发系统为应用、训练和测试网络设定参数,但不支持图形环境。“处理”的概念可简单地表示为几个网络的连接,但目前还不能对处理进行自动连接。在文本窗口编辑神经网络的处理对象时,应输入要读取和写入的文件名。通过C语言库函数可以将神经网络功能编入其他应用系统中去。

(6)CaseNet 神经网络开发环境

CaseNet 是一种应用于个人计算机的神经网络开发环境,如果神经网络软件能合理地设计并加以优化,大多数类型的问题就能在个人计算机上有效地解决。CaseNet 提供了一个图形界面,设计者可在屏幕上画出网络结构并通过屏幕上的菜单输入网络特性。网络图形、特性及图上的节点构成了神经网络的特征描述,由此可自动生成可执行的编码。CaseNet 的特点是将用户定义的网络特性生成高度优化的机器码,以便最大限度地利用现有个人计算机的功能。

CaseNet 系统由网络定义器、分析器、编码生成器和编译器4个主要工具组成。网络定义器是设计网络结构的图形编辑工具,后处理工具将用户设计的网络图形翻译成一种标准表达形式。分析器确认网络的定义并产生代码段。编译器将代码段和通用的网络框架编译成可执行的神经网络。在神经网络生成过程中,以用中间状态文件保存各阶段的结果,以使用户修改和优化网络各阶段的内容。

3.6.4 神经网络的硬件实现

1. 概述

神经网络的硬件实现是神经网络研究的基本问题之一。从对神经网络进行理论探讨的角度,可以通过计算机仿真途径来模拟实现特定的神经网络模型或算法,但在构造神经网络的实际应用系统时,必然要研究和解决其硬件实现的问题。神经网络专用硬件可提供高速度,并具有比通用串、并行机高得多的性能价格比,所以,特定应用下的高性能专用神经网络硬件是神经网络研究的最终目标。

神经网络本质上是一个并行分布式信息处理系统,其硬件实现极其复杂;神经网络结构上以大量的神经元单元为基础,每个神经元只需对输入信息进行加权求和、阈值运算等简单操作;神经网络运行中将记忆信息分布表示在神经

元间的连接权上;在神经网络的连接机制模型中,没有传统计算机中的 CPU、主存之类的概念,而是采用分布式信息存储和处理。

从大脑神经系统的结构组织上来看,所有神经元构成的系统具有串并联结构;从传统的计算机信息处理能力上来看,在智能信息处理(如知识的获取、知识的存储记忆、运用知识解决实际问题等)方面,显得能力相当有限,而这方面正是神经计算机所擅长的。神经网络理论研究为神经计算机的实现提供了体系结构基础;而微电子学、光学技术、生物工程技术、计算机科学技术为其物理实现提供了物质基础和手段。基于神经网络的结构模式和信息处理方法的神经计算机,标志着计算机及信息领域的一次根本意义上的技术革命。

(1) 主要研究内容

所谓神经网络的硬件实现,是指神经网络中的每一个神经元及每一连接都有与之相应的物理器件。

首先,神经元的超大规模集成电路(VLSI)实现是基础性内容。由于人工模型神经网络是由大量神经胞体通过特定形式的加权网络连接组成,因此可以认为神经网络由两种基元构成,即收集信号并完成非线性变换的神经胞体和完成各神经胞体间加权互连的突触。研究神经元的 VLSI 实现也就是要研究大量突触、神经胞体的 VLSI 集成以及突触和胞体间数据通信结构的实现。

在此基础上,还要研究神经网络的 VLSI 实现。在单个芯片中集成多个神经胞体和大量的突触单元,并将它们按某种通信结构组成神经网络系统。

基于各种神经网络模型的构造与实现,开展对于大规模多处理机并行的神经计算机体系结构的设计和实现的研究,已成为一个边缘技术领域。除了在现有的常规计算机上进行纯软件模拟,探讨能更好地支持神经网络实现的并行体系结构以外,借助 VLSI 技术制作神经网络协处理机、并行处理机阵列,无疑会大大推动神经网络的发展和应用。

其次,性能评价是神经网络硬件实现的重要研究内容。通过包含可实现模型的个数、物理处理单元的个数、容量和速度等一组指标,来衡量大规模神经网络实现的技术性能。

(2) 目前状况

随着现代 VLSI 工艺和技术的发展,目前已可以在单个芯片上集成几百万

个门电路,并已制造出单片计算机,单片专用系统,因而 VLSI 集成电路被认为是神经网络硬件实现的一种有效方式,有关学者已在这方面做了大量的工作。具有更高集成度的 VLSI 新工艺正在发展中,利用圆片集成技术或三维 VLSI 集成技术也可以构造更大规模的神经网络系统。若网络规模要求更大时,还可将多个芯片以积木式结构在板极上系统实现。

神经网络的 VLSI 实现方法可大致分为 3 类,即数字 VLSI 实现、模拟 VLSI 实现以及数模混合 VLSI 实现。

数字 VLSI 实现方法是采用二进制数字电路作为其基本运算模块和存储模块,它的优点是:技术成熟,测试方便,有大量的自动设计辅助工具,便于与新工艺接轨;权值存储简单、多样化,且权值学习方便;可以实现任意的精度;对噪声、窜扰、温度效应及电源波动具有较强免疫力;扩展性好,多个芯片易于连接成更大规模的网络。缺点是:用于实现突触连接的数字乘法器占芯片面积大,使大规模网络的单片集成很困难;大规模数字电路必须是同步的,而实际的神经网络是异步的;所有数值都要进行量化处理,而实际神经网络的状态和激励信号都是模拟值;速度比模拟 VLSI 电路低;单个信号值要用多个二进位表示,因而信号的通信机制复杂。

模拟 VLSI 实现方法是采用模拟电子电路作为基本运算模块和存储模块,它的优点是:自动地具有异步特性和平滑的神经激励;功能模块(如乘法器)占用芯片面积少,因而具有更高的集成度;运算速度快,且模拟信号的传输速度也快;器件的非线性效应便于非线性函数的实现。缺点是:工艺复杂,特别是为实现持久性权值存储及权值学习电路,需要更复杂的工艺;精度低,免疫力差;不同芯片中的电路参数差异可能较大,因而其扩展性差;只有有限的设计灵活性。

由于目前已有数模兼容的 VLSI 工艺,可将数字电路和模拟电路集成在同一个芯片中,因此又出现了神经网络的数模混合 VLSI 实现方法,它试图利用数字和模拟电路两者的主要优点,例如权值的存储和调整采用成熟的数字技术,运算单元采用快速模拟电路,有时为了与外部数字系统的接口兼容性,芯片的输入、输出采用数字信号,在芯片内再变换为模拟信号进行处理,通过数/模、模/数变换器完成数据形式的转换。各个模块是选用数字电路还是模拟电路,要综合考虑系统的各项性能要求,如速度、精度、集成度、可实现性以及实际应

用需要等。

无论是采用哪一种 VLSI 实现方法,都要遇到信号的运算、存储和传输这 3 个主要问题。神经网络中的基本运算主要有以下几种:①突触加权乘法;②神经元输入激励信号的累加;③阈值处理;④非线性函数运算;⑤权值学习和调整(其中既有较简单的 Hebb 学习规则、感知器学习规则,也有较复杂的“胜者取全”学习规则、模拟退火等)。信号存储主要是突触权值的存储,信号传输主要是神经元状态信号及误差信号的传输。

在数字 VLSI 实现方法中,乘法和累加运算可分别由数字乘法器(或标量积数字电路)和数字加法器(或数字计数器)来完成,有时为简化硬件,将信号值通过随机脉冲产生器变换成随机脉冲序列,这时乘法和累加运算分别由与门和或门就可实现。阈值处理可用数字比较器实现。较复杂的非线性函数运算一般是采用查寻表的方法实现;非线性函数还可分段线性函数来近似,从而可用简单的标量积数字电路来实现。权值的存储可采用 DRAM、SRAM、EPROM、ROM。信号的传输可采用与数字计算机中相同的技术,如总线结构、并行、流水结构等。

在模拟 VLSI 实现方法中,固定权值可用简单的电阻网络来实现;可调权值是用面结型或 MOS 型场效应管、CMOS 开关、开关电容电路、开关电阻器、跨导放大器等来实现;权值除用阻性电路实现外,一般还要用容性电路来存储,如 MOS 电路负载上的单电容、双电容(动态差值存储)等。乘法运算可用吉尔伯特(Gilbert)乘法器(包括各种改进型)、跨导放大器、基于高增益运放的乘法电路、开关电容电路、乘法 D/A 变换器等实现。加法运算一般是利用电流的汇集或电荷的累积来实现。阈值处理可用基于运放的模拟比较器实现。S 型非线性函数可用级联反相器、差分电压放大器、跨导放大器、开关电容等电路来实现。权值调整一般是通过改变场效应管的导通电阻、模拟放大器电路的增益、电容中的电荷量;权值学习电路与学习规则有关,即按照相应的学习规则具体设计特定的模拟电路,得出与权值调整量对应的电压或电流值,作为调整权值的控制量。模拟 VLSI 实现方法中的信号传输很直接,相应的模拟值将以电压、电流的形式在电路网络中流动。

日本、美国等已用现有成熟的 VLSI 工艺技术生产出电子神经元器件,以

及专用的具有一定功能的电子神经计算机。但是,随着神经网络规模的增大,神经元之间的高密度连接造成 VLSI 布线工艺上的困难和障碍。因此,所实现的网络规模受到很大局限。

(3) 发展前景

光子学的崛起是 20 世纪后期高科技领域的重大事件。近十年来光子技术以它优异的技术特色和强大的生命力,已在光通信、光存储、光集成、光互连、光计算、光信息处理和光神经网络等重要领域得到了广泛的应用并已获得令人瞩目的进展。

光学技术是实现神经网络及神经计算机的一个比较理想的选择。因为光学技术具有非常可贵的固有特性,主要体现在如下几方面:高度连接性,其光线扇入扇出系数大;有较高的通信带宽;以光速并行传递信息;光信号间无干涉性。虽然光学神经计算机实现技术目前还不是很成熟,其商品化大规模实现还有待时日,但一些光学神经元器件、光电神经计算机研究已有了较为迅速的进展,表现出广阔的发展和应用潜力,并引起相应领域的充分关注。

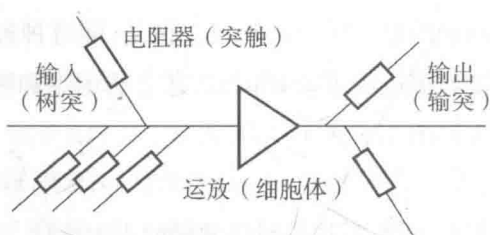
2. 神经元器件

神经元器件是构造神经网络硬件系统的基础,它包括突触电路、胞体电路以及实现某些学习规则的电路。VLSI 技术的迅速发展为神经网络的硬件实现提供了条件。神经网络的 VLSI 实现主要包括模拟实现、数字实现、模数混合实现及光电技术实现等几种形式。

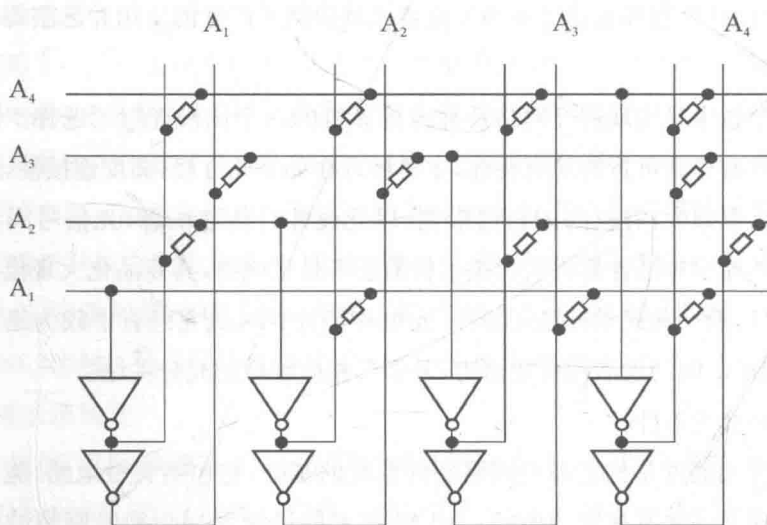
(1) 模拟式神经元器件

神经网络实现的基本单元是神经元器件,它由细胞体、树突、轴突和突触电路四大部分组成。作为神经元器件,必须具有如下功能特征:

- ①是一个多输入单输出的处理单元;
- ②具有加权求和能力;
- ③具有阈值处理或非线性函数处理能力;
- ④突触权值是可调节的。



(a) 单个神经元结构



(b) 互连神经元

图 3-22 神经网络硬件结构

用模拟电子技术实现的神经元器件,称为模拟神经元器件。通常,用模拟运算放大器代表细胞体,导线代表树突和轴突,电阻器代表突触连接,其阻值即为连接权值。典型的 Hopfield 网络的模拟神经元就是这样实现的。图 3-23 (a)为一个模拟神经元的结构,图 3-23 (b)为多个模拟神经元互连组成的神经网络硬件结构,它是一个全互联联想记忆模型的实现。图中,水平线为输入,垂直线为输出。每个神经元器件的输出通过一个电阻器与其他神经元器件输出相连。

神经网络的学习功能是通过改变突触电路的权值来实现的。如何实现突触权值的变化是设计和实现电子神经元器件的关键。在模拟电路的实现技术中,主要是通过控制 MOS 晶体管的导通电阻,控制导通晶体管的数目,采用

RAM 电路以及采用高集成度的薄膜技术(包括非晶半导体技术、金属氮氧化物半导体 MNOS 技术)来改变权值。

神经网络的实际应用依赖于神经元器件的集成化实现,即将大量神经元集成在单一 VLSI 芯片上。现有的电压模式 VLSI 技术,难以获得精确、稳定的集成电阻,且占有很大的芯片面积,使大量神经元集成化遇到巨大的限制。研究表明,神经网络结构特性表明,突触连接具有跨导特性。因此,应用基于电流模式 VLSI 技术的跨导放大器,为神经元器件的集成化实现开辟了一条新的途径,跨导放大器有望成为模拟 VLSI 实现的神经元器件。电流模式 VLSI 技术的发展打破了电压模式一统天下的格局,形成新的技术应用格局。

(2) 数字式神经元器件

由数字逻辑电路实现的神经元器件,称为数字神经元器件。图 3-24 所示为数字神经元原理图,它由突触电路、树突电路和细胞体电路 3 部分组成,神经元轴突的功能是输出信号,可简单地采用输出导线实现。突触电路通过调节输入脉冲密度,实现权值变化;树突电路由简单的逻辑“或”门构成,对来自各突触电路的脉冲进行空间求和。无论是兴奋脉冲还是抑制脉冲都具有相同的极性,因此,每个神经元采用两个树突电路,以示区别。兴奋性树突电路与抑制性树突电路分别与细胞体电路的上/下计数器的输入端相连,由细胞体电路产生输出。这时认为神经元的输入和突触权值都为二进制数字信号,神经元将完成下面的功能:

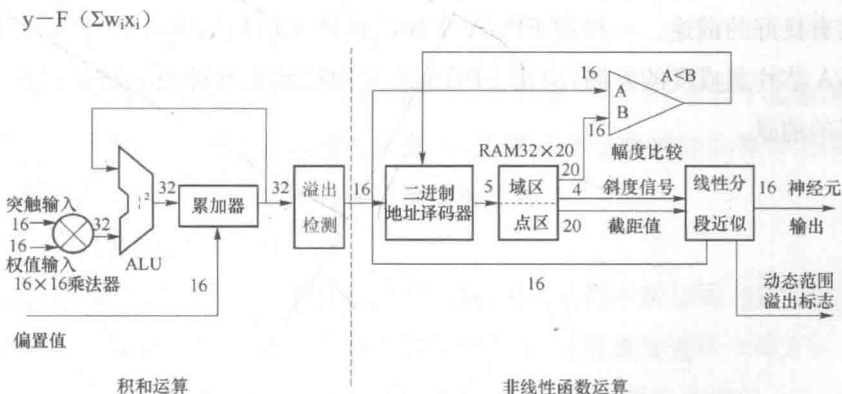


图 3-24 数字神经元原理图

对应的神经元电路原理图如图 3-25 所示。图中,数字乘法器完成 $X_i = w_i x_i$ 运算,加法器完成 $X = \sum x_i$ 运算,非线性函数变换模块完成 $y = F(X)$ 运算。

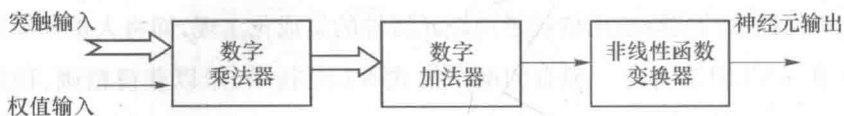


图 3-25 神经元电路原理图

这种形式的神经元电路实现相对容易,数字乘法器和数字加法器的实现电路已有很多种,这里不做详细介绍。应该指出的是:神经网络的运算对精度要求不高,一般认为 16 位的定点运算已足够,所以数字乘法器和加法器的实现电路应力求简单。简单的阈值处理可用数字比较器实现,较复杂些的函数大多以查寻表的方式实现。作为数字信号处理模块的神经元电路也可以将学习电路结合到一起,但电路结构将更为复杂。

(3)FPGA 神经元结构

FPGA 是现场可编程门阵列的简称,它的制造工艺与一般的 VLSICMOS 门阵列一样,不同的是它具有用户可编程性。它主要由两部分组成,一个是基本胞元阵列,担负各种逻辑运算和信息存储功能;另一个是开关格阵,完成基本胞元间的互连结构。市场上出现多种系列的 FPGA 芯片产品。

FPGA 芯片的特点是:电路简单,占据芯片面积小;运算速度快,造价低;易于用户编程设定基本胞元功能和开关格阵的结构,灵活性大;设计周期短,对设计改进方便;可靠性好,保密性强。这些特点决定了用 FPGA 构造数字神经网络具有良好的前途。一种用 FPGA 实现的神经元结构如图 3-26 所示。随着 FPGA 芯片集成度的发展,应用 FPGA 芯片和技术实现神经元结构具有非常良好的前景。

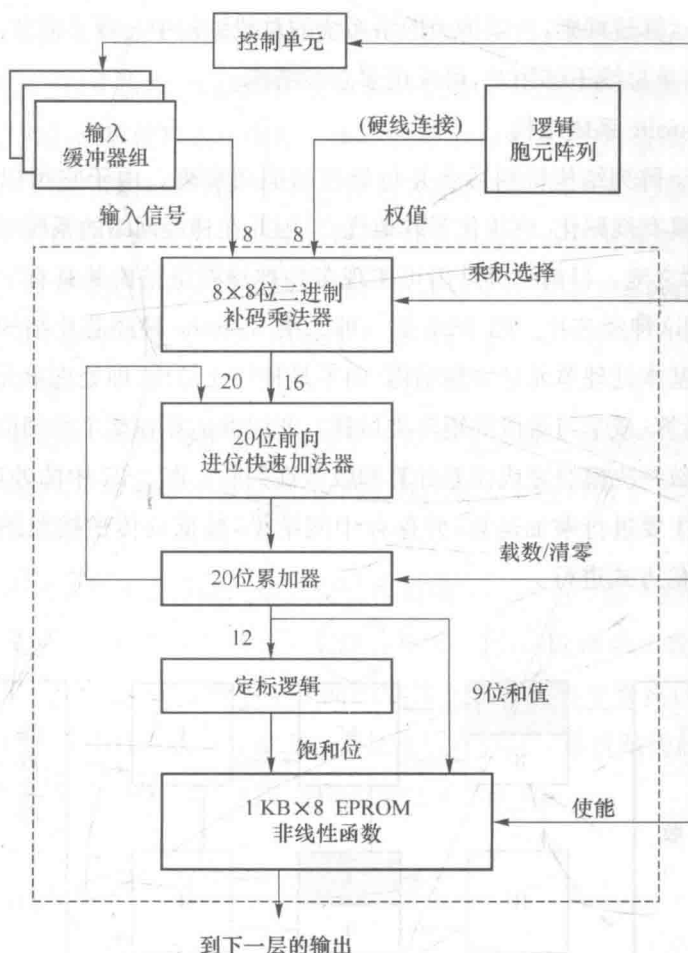


图 3-26 一种用 FPGA 实现的神经元结构

3. 神经网络系统结构

神经网络是由大量神经元组成的网络,为在单个 VLSI 芯片中实现神经网络系统,就不仅要集成大量的神经元器件,还要完成这些神经元器件之间的互连。依连接方式不同,形成了各种系统结构形式。

(1) 总线连接系统结构

神经网络系统结构中的总线类似于多处理器结构中的总线,但它要完成的任务更艰巨,因为神经网络中有更多的处理单元,且经常要进行全局互连,若神经网络中有 n 个节点,则其连接复杂度为 $O(n_2)$ 。由于每个处理单元中的运算

时间较长,运算较规则,所以神经网络系统的总线结构中一般不需要高速 cache 缓冲器。当单总线不够用时,可采用多总线结构。

(2) Systolic 系统结构

Systolic 阵列结构特别适合并行处理的应用实现。由于它可以获得线性加速比,并具有规则化、模块化等其他优点,因此在神经网络的系统结构设计中占有一席之地。目前已设计出可实现多种神经网络结构并具有学习能力的通用 Systolic 神经芯片。图 3-27 是一种二维 Systolic 神经芯片结构。这种系统结构中,基本处理单元是突触结构,而不是神经元结构,即处理单元阵列主要用于积和运算(或学习阶段的矩阵类运算),非线性运算在单个的阈值处理模块中进行,收敛判决模块完成误差计算和收敛性判断。图 3-27 中的处理单元(突触结构)也主要进行乘加运算,并保存中间结果,数据的传送按照图中所示的 Systolic 通信方式进行。

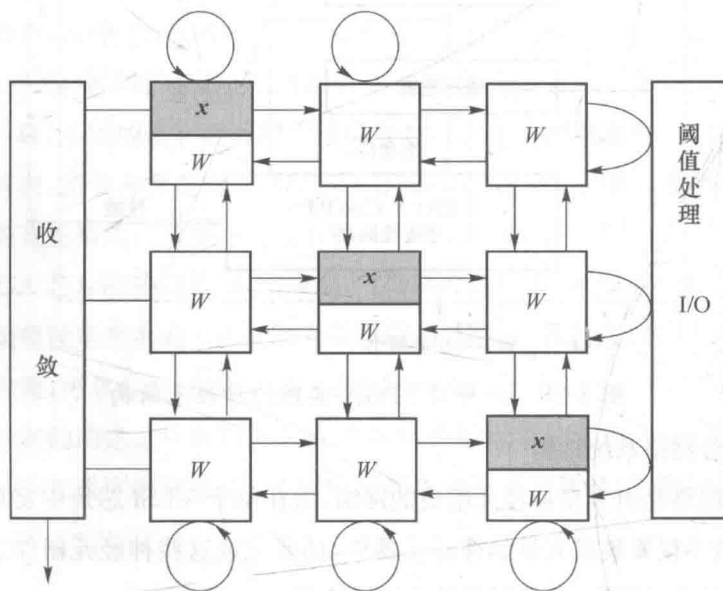


图 3-27 二维 Systolic 神经芯片结构

随着 VLSI 工艺的进步,或对突触结构进行简化,Systolic 阵列结构在神经网络系统实现中大有前途。

(3) 树流水式系统结构

数字 VLSI 的集成度预计到 21 世纪末最多只能再提高 100 倍左右,即在

单个芯片上最多只能集成约 5000 个乘法处理单元。由于这种物理限制,神经网络到数字 VLSI 芯片中的集成就存在两种趋势,一种是将上述的处理单元看作是一个神经元,以使得能在单片上集成数千个神经元,但存在有多突触神经元速度低,通信控制复杂等问题;另一种趋势是将处理单元看作是一个突触结构,即用多个处理单元实现一个神经元,这时当神经网络规模增大(即神经元增多,相应于每个神经元的突触也增多)时,集成的系统速度不会降低,且实现结构与神经网络模型更接近,但它所能单片实现的神经元个数少,且需要有很高的通信带宽的互联结构。

树流水式神经网络系统结构的基本出发点还是神经突触到处理单元的映射,但这种结构可以很方便地缩减,以使实现的神经元数日增多,系统所需的通信带宽减少。

最基本的树流水式神经网络系统结构如图 3-28 所示,每个神经元都用一个内乘加单元组成的树流水结构来实现。显然,所需的处理单元数是 $O(n_2)$ 量级(假定神经元数为 n),且处理单元间的直接互联将需要非常高的通信带宽。但并行处理中的多任务流水技术可以灵活地运用到这一系统结构的简化中,即让图 3-28 中的一个树流水结构完成多个神经元的运算。

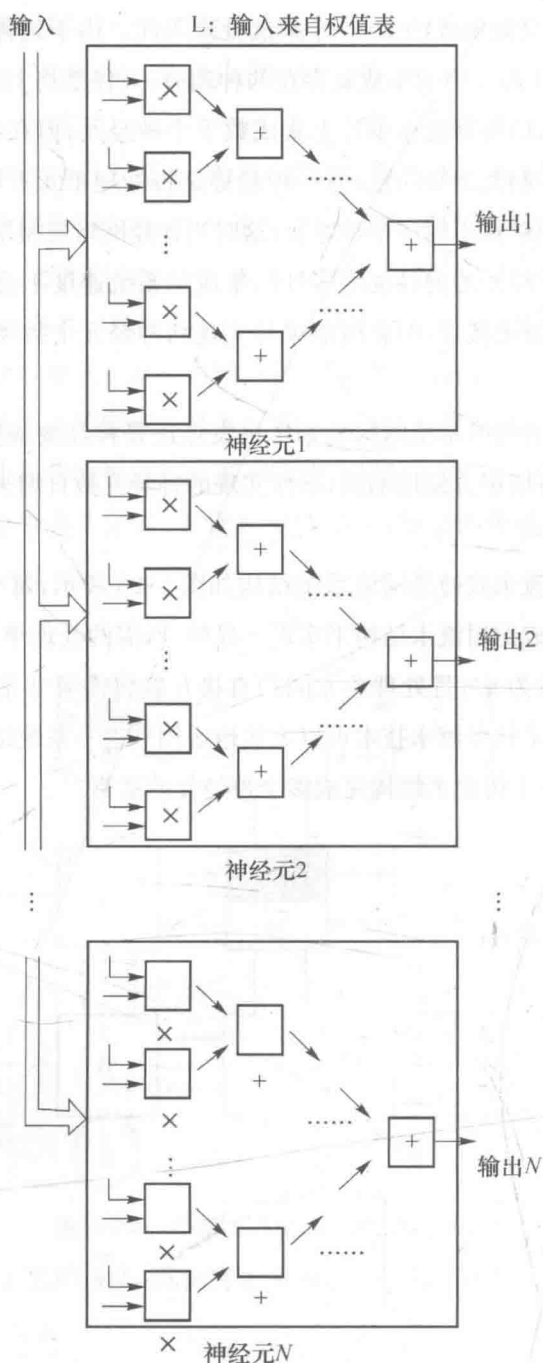


图 3-28 树流水式神经网络系统结构

(4) 自组织类系统结构

采用竞争学习规则的自组织神经网络是一类重要的神经网络,其中包括 Kohonen 网络, Grossberg 网络(ART), Fukushima 网络(neocognitron)等。这类神经网络所涉及的一些运算往往与其他的一般神经网络不同,因此它的系统结构实现也有其独特性。

以 Kohonen 的 LVQ 算法为例。该算法中的“胜者取全”部分运算可简单地表述为:

$$\min\{\|X_1 - X\|, \|X_2 - X\|, \dots, \|X_k - X\|\}$$

其中, $X_1 \sim X_k$ 为 k 个权值向量, X 是输入向量, $\|X_i - X\|$ 表示向量 X_i 和 X 的差向量的范数。

可用图 3-29 所示的结构来实现。图中,相应于 k 个权值向量的范数计算用 k 个范数处理器来完成(图中假定 $k=64$),最小值的检测是用一个二叉树搜索结构来实现的,它需要 $(2k-1)$ 个 min 处理单元。

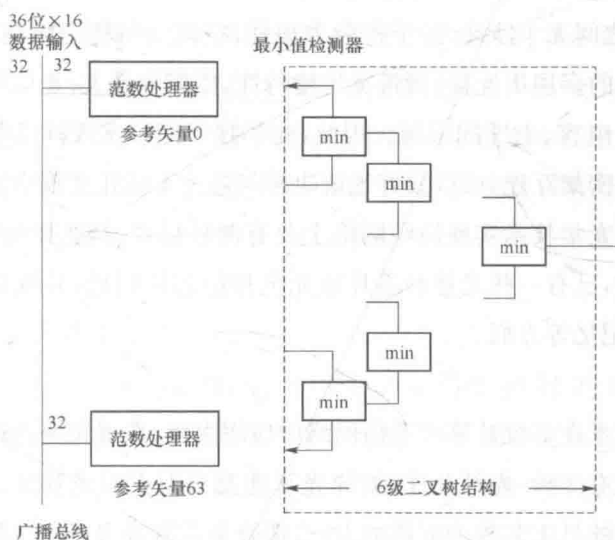


图 3-29 “胜者为王”运算结构

范数处理器是一个简单、通用的算术运算结构,可完成各类范数的计算,相应于向量各分量的运算将按流水方式顺序进行。已有实际的芯片单片集成了 64 个如图 3-29 所示的“胜者为王”电路。在片外再加上一个处理器完成 LVQ 算法的其他运算,就完整地实现了 Kohonen 网络。

4. 神经网络的光学实现

用光学或光子学方法实现的人工神经网络称为光学神经网络,相应地利用光学技术、光学器件或光电混合器件实现的神经计算机称为光神经计算机。从计算机发展的角度来看,神经网络与光学技术是两个极有前景的研究分支。基于二者技术相结合所产生的光神经计算机,是神经网络全硬件实现的一个重要形式。

(1) 光学技术与神经网络

光学技术与神经网络的关系之所以比较密切,是因为这门技术在神经网络全硬件实现上有其独到的优势。神经网络所需要完成的主要运算,用光学器件很容易实现。

与光学技术相比较,用电学方法所能实现的神经元芯片在规模上是相当有限的。随着神经元数目的增加,神经元之间的连接数目急剧上升,从而引起无法解决的高密集度的电子布线问题。而以光学信号作为信息传递的媒介有如下优点:信号之间无干涉性,抗干扰能力很强;空间分解能力很高,容易实现神经网络所需要的多扇出连接;高带宽传输特性,传播容量大;光信号通过介质传播,不受电阻、电容、电感的限制。因此,光学技术能够实现神经网络中大量神经元之间的高密集互连,可以较好地解决神经芯片集成化实现中遇到的电子布线难题。利用光学技术实现神经网络主要有两种形式:纯光学的、光学与电学混合的。目前,已有一些光神经芯片或光电神经芯片问世,并成功地应用于模式识别、联想记忆等方面。

① 光互连

光互连技术在传统计算机通信网络中应用多年,发展相当迅速。目前实用的光耦合形式有3种:光纤互连、波导光互连及自由空间光互连。光纤互连技术是一种可靠的且工艺较为成熟的方法,适合于在有较大互连空间的情况下使用;波导光互连技术可用来实现芯片内部或晶片之间的连接;自由空间光互连技术最为实用,它使光源发出的光束通过聚焦光学元件(如光学透镜阵列、全息元件等)成像或聚束后,集中照射到光检测装置上。

② 全息技术

最有希望建立任意光学互连的器件是全息图。所谓全息图是用于建立任

意光互连的一个光束控制阵列。众所周知,全息技术是产生三维图像的重要手段之一。然而,从更一般的角度来看,它又是一种能记录光线的强度和入射方向的存储装置。常规光学透镜只是把入射光线投射到像平面的某一特定位置,而全息图则很容易对之加以“编程”,产生许许多多类似的投射。全息图又分为平面全息图和体全息图两类。

平面全息图。平面全息图是在比较薄的介质(如照相胶片)上制作的,它能够使位于它的某一侧的任意一束光射到另一侧(或同一侧)的任意一点上,只要点与光束的互连总数不超过胶片上可分辨点的总数。据统计,1平方英寸(约 66.45cm^2)的全息图上可分辨点的数目可达1亿之多。这样的全息图可把104个点光源中的每一个光源都同上万个光检测器中的每一个完全互连起来,而用电子布线方式在硅片上完成类似的互连则是极其困难的。

体全息图。用光折射晶体制成的体全息图,其光互连能力更令人吃惊。当光折射晶体受光照射时,晶体内就产生了按光强度而分布的电荷。电子光折射晶体中的局部电荷密度决定了局部折射率(某物质的折射率是光在该物质中行进速度的度量),所以,投射到晶体上的全息图像就以空间变化的折射率的形式记录下来。以后,只要用光束照射该晶体,就能从全息图中获取记录的图像信息。体全息图将互连信息存储在三维空间中的能力,为光神经计算机提供了一种潜在的巨容量存储器。从分辨率来讲,体积为 1cm^3 的全息图能够实现的连接数多于1万亿个。

(2) 光学逻辑器件

一种光学材料,如果其透射或反射特性(不透明度、透射率、反射率等)随投射到其表面的入射光的亮度的变化而呈非线性变化,则称之为非线性的。砷化镓、硫化锌、硒化锌等均为非线性光学半导体材料。同硅材料相比,这种光学材料的主要特点是:响应速度快、高辐射阻抗、工作温度范围宽、适合于苛刻环境。

① 光学逻辑门

光学逻辑门是由非线性半导体材料制成的。当用一束或多束聚焦的单色光照射半导体材料时,入射光强与透射光强的关系曲线如图3-30所示。由此可见,光学逻辑器件具有双稳态特性。逻辑“0”相应于 $I_1 \sim I_3$ 段的入射光强,逻辑“1”相应于 $I_2 \sim I_4$ 的入射光强。

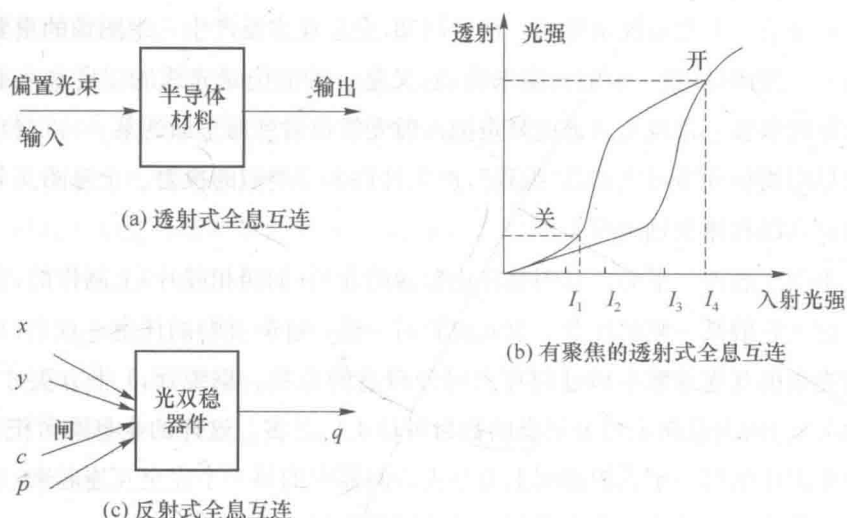


图 3-30 光逻辑门双稳态特性

光双稳逻辑门能够实现二值逻辑效应,如完成逻辑“与”“或非”运算、存储功能等。对于一个二输入的光双稳逻辑器件,假设输入记为 x 和 y ,控制信号 c 决定器件功能,权输入 p 建立入射光强基值,那么相应的输出 q 的逻辑方程表达为

$$q = \bar{c}q + c(x + y)$$

$$\bar{q} = \bar{c}q + c(\overline{x + y})$$

这说明该器件可用作逻辑“或”门或逻辑“或非”门(当控制信号 c 为 1 时);或者作为纯粹的存储器件(当 c 为 0 时)。如果用两束输入光,一个作偏置输入,另一个作控制用,那么与输出光束一起便可构成光晶体管。

② 光计算和光存储

光在传递信息的同时,也可以对信息进行加工处理,这就是光学信息处理或光计算。对于人工神经网络的光学实现来说,光计算所固有的特性是非常重要的,它不但具有完成线性变换(如傅里叶变换、二维卷积和相关、矩阵相乘等)和非线性变换(如双稳态、逻辑运算等)的能力,并且具有在信息处理过程中难得的高度并行性和高速性特点。很多光计算和光学信息处理系统,实际上都可看作是一种特殊的光学神经网络。特别是近年来在光计算和光学信息处理中,有一种引入非线性和反馈迭代过程的趋势,这和光学神经网络研究几乎是殊途

同归。因此,可以肯定地说,光学信息处理和光计算的已有进展都将或多或少地在神经网络的光学实现方法研究中得到应用和发展。

光存储是指用光学手段实现信息存储的技术。光存储的主要优点是信息存取(或写读)的并行性和巨大的存储容量,这正是神经网络的硬件实现所要求的特性。常见的光存储器有2种。

a. 感光片。其二维信息存储能力可有 106 bit/mm^2 ,但是存储时间受化学处理过程(显影、定影)的限制,不能实时,也不能擦除。

b. 光盘。近年来已经商品化的光学存储器,并已成功地用于音乐、图像的存储和计算机外设存储器,并行存取的可擦除光盘是光学神经网络的首选存储技术之一。

全息存储是光存储技术中最具特色的。全息存储的信息容量比感光照相要高得多。一般全息片的信息存储密度可高达 109 bit/mm^2 。全息存储器在很多方面与神经网络的联想记忆有相似之处,并具有良好的鲁棒性。全息存储的另一个特点是它可以把存储与处理结合在一起来进行。不同的全息存储材料在记录的光谱范围、灵敏度、分辨率、响应速度及存储时间等方面有不同的性能。理想的全息记录材料的研究是光存储技术中的关键问题,这一问题的解决不仅会大大推进光学神经网络的研究,也会对整个信息技术产生重大的影响。

(3) 光神经网络的研究现状

20世纪80年代初期,随着神经网络的重新兴起,光神经网络的进展也十分令人瞩目,带动光神经计算机的迅速发展。目前,光神经计算机的研究和实现已经取得了许多卓有成效的结果。无论是基于传统计算机的神经网络开发环境,还是神经网络协处理加速装置、神经网络并行处理机阵列,它们为神经网络的研究和应用提供了一个基本的工作平台。

光学器件构成的神经计算机,其基本结构框架包括输入/输出接口、存储单元、处理机阵列及互连。输入/输出单元是信息传送器,将数字或符号数据转换为二维光学数据,并显示处理结果数据。典型的器件如空间光调制器。存储单元是信息储存的中心,一般包括两部分:主存和数据库。主存由体全息图及空间光滤波器组成;数据库使用光盘实现。处理器阵列是一个并行逻辑门阵列,由光电器件或空间光调制器实现,组成一个处理环。互连是光神经计算机的优

势所在,互连器件可以是体全息图、光纤、透镜等。

光神经计算机自然地分为两类:模拟式的和数字式的,这与传统电子计算机是类似的。模拟光神经计算机通常用来完成突触权矩阵与输入向量的乘积,其核心部分是模拟光处理器。数字光神经计算机是由光学双稳器件来实现的。在美国、日本、西欧等国,越来越多的光神经计算机或光电神经计算机装置相继问世。标志着人们已经向真正的光神经计算机迈进了重要的第一步。对于神经计算机这一新兴的计算机学科分支,仍然有许多难题需要探索。在理论研究方面,最重要的是神经网络学习、自组织动力学的研究,进一步探索复杂神经网络的非线性动力学及其功能。从已有的神经网络模型和应用来看,人们对神经网络动力学系统还缺乏全面深入的研究;神经计算机的理论体系有待于完善和发展。在技术研究方面,为了使神经计算机实用化,必须研究具有高性能的神经芯片。高度集成化和高速化是神经芯片性能的重要指标,以此推动光神经芯片和光神经计算机研究的进展。

第 4 章 主流的机器学习模型

4.1 树模型

树模型是机器学习中最为常见的模型之一。例如, Xbox 游戏机的 Kinect 运动感知设备中的姿态识别算法, 其核心便是决策树分类器。树模型具有较强的表示能力, 易于理解, 且因其递归的“分治”本质而尤其受到计算机科学研究人员的关注。

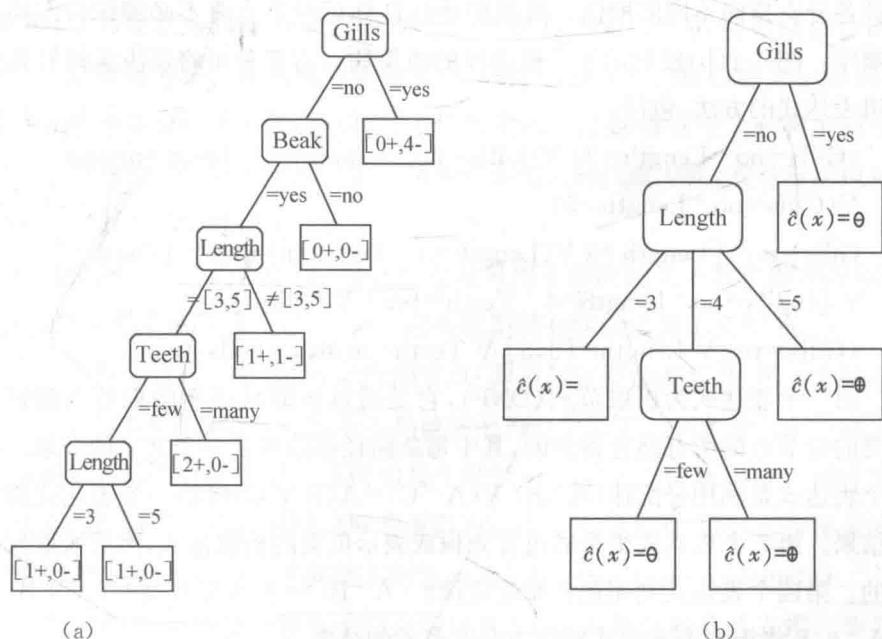


图 4-1 (a)以树形式表示的路径;(b)学习到的决策树

例如, 如图 4-1(a)所示的特征树。

(1) 特征树中最左端的叶节点表示路径最底部的概念, 它覆盖了一个正实例。

(2) 沿该路径向上的下一个概念通过内部析取将文字 $\text{Length}=3$ 推广为 $\text{Length}=[3, 5]$; 所增加的覆盖(一个正实例)为该特征树左端第二个叶节点所表示。

(3)将条件 Teeth=few 丢弃,可增加另外两个被覆盖的正例。

(4)将条件“Length”丢弃,可增加最后一个正例及一个负例。

(5)将条件 Beak=yes 丢弃并不会增加额外的覆盖。

(6)最后,将条件 Gills=no 丢弃,可覆盖剩余的四个负例。

我们看到,任何一条穿越假设空间的路径都可被转化为一棵等价的决策树。为获得一棵与该路径自最底端起向上的第 i 个概念等价的树,我们既可将最左端的 i 个叶节点合并为一个表示该概念的叶节点,也可将最左端的 i 个叶节点标记为正类,而将其余叶节点标记为负类,从而将特征树转化为决策树。

对于取值超过两个状态的特征,决策树并不会运用内部析取来处理,而是允许各分支指向不同的取值。决策树还允许标记叶节点时不必遵循自左向右的顺序。图 4-1(b)便展示了一棵这样的决策树。有多种可将该决策树转化为逻辑表达式的方法,包括:

$(\text{Gills}=\text{no} \wedge \text{Length}=3) \vee (\text{Gills}=\text{no} \wedge \text{Length}=4 \wedge \text{Teeth}=\text{many})$

$\vee (\text{Gills}=\text{no} \wedge \text{Length}=5)$

$\text{Gills}=\text{no} \wedge [\text{Length}=3 \vee (\text{Length}=4 \wedge \text{Teeth}=\text{many}) \vee \text{Length}=5]$

$\vee [(\text{Gills}=\text{no} \wedge \text{Length}=4 \wedge \text{Teeth}=\text{few}) \vee \text{Gills}=\text{yes}]$

$(\text{Gills}=\text{yes} \vee \text{Length}=[3,5] \vee \text{Teeth}=\text{many}) \wedge \text{Gills}=\text{no}$

第一个表达式为析取范式(DNF),它是通过析取从该树的根节点到标为正类的叶节点的所有路径得到的,其中每条路径都给出了一个文字的合取。第二个表达式是利用分配律 $(A \wedge B) \vee (A \wedge C) = A(B \vee C)$ 将第一个表达式简化的结果。第三个表达式则是通过首先构成表示负类的析取范式,然后再取反得到的。第四个表达式利用德·摩根定律 $(A \wedge B) \equiv \neg(A \vee B)$ 及 $\neg(A \vee B) \equiv \neg A \wedge \neg B$ 将第三个表达式转化为合取范式的结果。

此外,还有许多其他逻辑表达式与由决策树定义的概念等价。那么是否可能得到等价的合取概念?有意思的是,这个问题的答案是否定的:虽然有些决策树可表示合取概念,但许多决策树却无法表示,前面那个决策树正是其中之一。决策树具有比合取概念(严格)更强的表示能力。实际上,由于决策树对应于析取范式,且每个逻辑表达式可等价地表示为析取范式,因此决策树具有最强的表示能力。只有一类数据是决策树无法应对的,即那些标注不一致(相同实例以不同标签出现两次)的数据。这解释了数据不能以合取方式分离却可

被决策树分离的原因。

使用这样一种具有较强表示能力的假设语言时,存在一个潜在的问题。令 A 表示所有正例的析取,则 Δ 为析取范式。显然, Δ 覆盖了所有正例——实际上, Δ 的外延正是所有正例构成的集合。换言之,在析取范式的(或决策树的)假设空间中, Δ 为正例的 LGG,但不会覆盖任何其他实例。因此, Δ 难以推广到正例之外的情形,而只是记忆了这些正例——没错,过拟合!由此我们想到,有意选择一种具有一定局限性的假设语言是一种避免过拟合且有利于学习的方式,例如合取概念。使用这样的语言,即便是 LGG 运算,通常也能推广到正例之外的情形。如果我们采用的假设语言足以表示任意正例构成的集合,则必须确保学习算法运用了某种可强制推广到现有实例之外以避免过拟合的机制——这被称为学习算法的归纳偏置(inductive bias)。我们将会了解到,大多数在表示能力较强的假设空间中工作的学习算法都具有一个有利于减小假设复杂性的归纳偏置——或是隐式调整假设空间的搜索方式,或是显式地在目标函数中引入复杂性罚项。

树模型并不只局限于分类场合,而是可用于解决绝大多数机器学习问题,包括排序、概率估计、回归及聚类。这些模型共有的树结构定义如下。

特征树是这样一种树,每个内部节点(即不为叶节点的节点)用一个特征来标记,且从内部节点出发的每条边都被标记为一个文字。一个节点中所有文字构成的集合称为一个分裂。该树的每个叶节点表示一个由从根节点到叶节点所遇到的所有文字所形成的合取的逻辑表达式。该合取的外延(它所覆盖的实例集)被称为与该叶节点关联的实例空间区隔(instance space segment)。

本质上,特征树是一种表示假设空间中大量合取概念的简洁方式。学习算法的任务则是确定哪些概念最适于解决既定问题。算法 4.1 给出了大多数树学习器共通的一般性学习步骤,其中用到的三个函数定义如下。

(1) Homogeneous(D) 若 D 中实例的纯度足够高,从而可用单个类别标签来标识,则返回真,否则返回假。

(2) Label(D) 返回对于实例集 D 最恰当的分类标签。

(3) BestSplit(D, F) 返回将要放置在根节点的文字的最优集合。

这些函数的具体定义依赖于给定任务。例如,对于分类任务,当大多数实

例都来自同一类别时,说明这组实例具有较高的纯度,因此最恰当的标签应取为多数类的标签;对于聚类任务,当一组实例彼此足够接近时,说明它们纯度较高,最恰当的标签可取为某个范例,如该组实例的均值。

算法 4.1 $\text{GrowTree}(D, F)$: 依据训练数据学习一个特征树模型

输入 数据 D ; 特征集 F

输出 叶节点带有标签信息的特征树 T

if $\text{Homogeneous}(D)$ then return $\text{Label}(D)$; // 关于 Homogeneous , Label 函数请参阅正文

$S \leftarrow \text{BestSplit}(D, F)$;

将 D 依据 S 中的文字分裂为若干子集

for each i do

if $D_i \neq ?$ then $T_i \leftarrow \text{GrowTree}(D_i, F)$ else T_i 为一标记为 $\text{Label}(D)$ 的叶节点

end

返回 根节点被标记为 S , 其子节点为 T_i 的特征树

算法 4.1 为一分治算法,它将数据划分为若干子集,并对每个子集构建一棵树,然后将这些子树合并为一棵树。分治算法是计算机科学中一项久经考验的技巧。通常人们会以递归方式实现这类算法,因为每个子问题(依据数据的某个子集构建一棵树)与原问题具有完全相同的形式。只要能够合理地设定递归的终止条件(通常位于算法实现的第一行),这种策略总是可行的。然而,应当注意,这类算法本质上属于贪婪(greedy)算法,在需要做选择的时候(如选择最优分裂),算法是基于当前可用信息做出选择的,而在做出选择之后,这一选择将不会被重新考虑。这类算法无法保证得到原问题的最优解。另一类策略是利用回溯(backtracking search)算法,该算法能够保证返回最优解,但会增加计算时间和内存开销为代价。限于篇幅,本书将不涉及此类算法。

在本章的其余部分中,我们会将算法 4.1 这个一般性算法实际应用到分类、排序、概率估计、聚类及回归等具体任务中。

4.1.1 决策树

如前所述,对于分类任务,当某个实例集 D 中的所有实例均来自同一类别

时,我们称为同质的(homogeneous)。显然,此时函数 $\text{Label}(D)$ 的返回值为中实例所对应的那个类别。在算法 4-1 的第 5 行,对于存在空子集 A 的非同质实例集,也可调用 $\text{Label}(D)$ 得到实例集 D 的标签。因此, $\text{Label}(D)$ 更一般的定义是返回实例集 D 中的多数类。接下来需要考虑如何定义函数 $\text{BestSplit}(D, F)$ 。

不妨假设所处理的特征为布尔型, D 可划分为子集 D_1 和 D_2 。同时假设共有两个类别,并用 $D_1+?$ 和 $D_1-?$ 分别表示 D 中的正例和负例所构成的集合($D_1+?$ 等的含义同上)。那么,如何对将样本划分为正类和负类的特征的有效性进行评价? 显然,最理想的情况是 $D_1+? = D_1+?$ 且 $D_1-? = ?$, 或 $D_1+? = ?$ 且 $D_1-? = D_1-?$ 。此时,就称这两个分裂的子节点为纯的(pure)。这样,我们便需要度量由 $n_1+?$ 个正例和 $n_1-?$ 个负例所构成的集合的杂度。我们需要遵循的一条重要原则是,杂度应仅依赖于 $n_1+?$ 和 $n_1-?$ 的相对大小,当将这两个量分别乘以相同因子时,度量结果不应发生变化。这就意味着杂度可依据比例 $p = n_1+? / (n_1+? + n_1-?)$ 来定义。此外,当交换正类和负类时,杂度的值也不应发生变化,这意味着当我们将 p 替换为 $1-p$ 时,杂度的值应保持不变。我们还希望当 $p=0$ 或 $p=1$ 时,杂度的大小为 0; 当 $p=1/2$ 时,杂度取得最大值。不难验证,下列函数均满足上述要求。

少数类(minority class) $\min(p, 1-p)$ 该函数有时被称为误差率,因为当叶节点被标记为多数类时,该函数度量的是被误分类的样本的比例;样本集的纯度越高,错误率便越小。该杂度指标可等价地表示为 $1/2 - |p - 1/2|$ 。

Gini 指标(Gini index) $2p(1-p)$ 该函数表示随机标记叶节点中的样本(每个样本被标记为正例的概率为 p , 而被标记为负例的概率为 $1-p$)所对应的期望误差。这样,假正例的出现概率便为 $p(1-p)$, 而假负例的出现概率为 $(1-p)p$ 。

熵(entropy) $-p \log_2 p - (1-p) \log_2 (1-p)$ 该函数刻画的是指明随机抽取样本的类别时所获得的期望信息量,单位为比特。样本集的纯度越高,样本类别的可预测性就越高,期望信息量也就越小。

这三种杂度所对应的函数曲线如图 4-2(a)所示,一些函数经过了缩放,以

使其能够在(0.5, 0.5)处达到最大值。此外,还增加了第四条曲线:Gini 指标的平方根,图中用标识。将看到,该指标与其他指标相比具有一项优势(即类分布的不敏感性)。若用表示叶节点的杂度,则一组互斥的叶节点所构成的集合 $\{D_1, \dots, D_l\}$ 的杂度可定义为各叶节点杂度的加权平均:

$$Imp(\{D_1, \dots, D_l\}) = \sum_{j=1}^l \frac{|D_j|}{|D|} Imp(D_j) \quad (4-1)$$

其中,对于二元分裂,给定父节点和子节点的经验概率时,存在一种求取 $Imp(\{D_1, D_2\})$ 的良好几何构造,如图 4-2(b)所示。

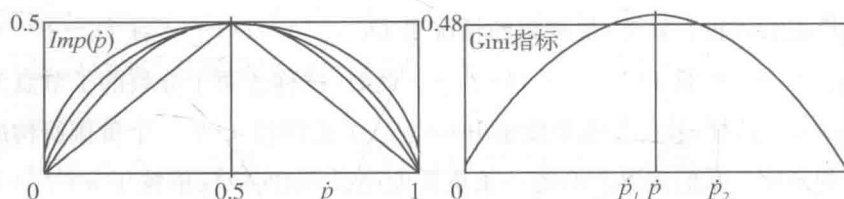


图 4-2 依据正类的经验概率所绘制的杂度函数

自底向上依次为:少数类的相对大小 $\min(p, 1-p)$; Gini 指标 $2p(1-p)$; 熵 $-p \log_2 p - (1-p) \log_2 (1-p)$ (绘制图像时将该函数除以 2,使其与其他函数在相同位置取得最大值);以及(缩放过的)Gini 指标的平方根 $\sqrt{p(1-p)}$ 。请注意,最后一个函数对应一个半圆。(b)用于确定某个分裂。

(1)首先求得杂度曲线(这里用的是 Gini 指标)上两个子节点的杂度 $Imp(D_1)$ 和 $Imp(D_2)$ 的值。

(2)然后用一条直线将这两个值相连,因为这两个值的任意加权均值都位于该直线之上。

(3)由于父节点的经验概率也为子节点经验概率的加权平均,其权重和之前杂度加权平均的权重相同,因此 p 给出了正确的插值点。

该构造适用于图 4-1(a)中所给出的任何杂度测度。需注意,当父节点类分布存在明显偏斜时,其两个子节点的经验概率可能会都处在直线 $p=0.5$ 的左侧或右侧。这并不什么问题,除了使用少数类的杂度测度外,因为从该测度的几何构造可以看出,所有位于同侧的分裂都会得到相同的加权平均杂度。因此,通常不建议采用少数类作为杂度测度。

4.1.2 作为减小方差的树学习方法

1. 回归树

在回归问题中,目标变量是连续型而非二值型的。在这种情形下,我们将一组目标值的集合 Y 的方差定义为 Y 中各元素到该集合均值的均方距离:

$$\text{Var}(Y) = \frac{1}{|Y|} \sum_{y \in Y} (y - \bar{y})^2 \quad (4-2)$$

其中, $\bar{y} = \frac{1}{|Y|} \sum_{y \in Y} y$ 为 Y 中目标值的均值。若某个分裂将目标值集合 Y 划分为一组互斥子集 $\{Y_1, \dots, Y_l\}$ 则加权平均方差为

$$\text{Var}(\{Y_1, \dots, Y_l\}) = \frac{1}{Y} \sum_{y \in Y} y^2 - \sum_{y=1}^l \frac{|Y_i|}{Y} \bar{y}_i^2 \quad (4-3)$$

数集 $X \in \mathbf{R}$ 的方差定义为该集合中元素到其均值的均方差:

$$\text{Var}(X) = \frac{1}{|X|} \sum_{x \in X} (x - \bar{x})^2 \quad (4-4)$$

其中, $\bar{x} = \frac{1}{|X|} \sum_{x \in X} x$ 为 X 中目标值的均值。展开 $(x - \bar{x})^2 = x^2 - 2x\bar{x} + \bar{x}^2$, 则上式可整理为

$$\text{Var}(X) = \frac{1}{|X|} \sum_{x \in X} x^2 - \bar{x}^2 \quad (4-5)$$

因此方差是集合中元素平方的均值与均值的平方之差。

有时,考虑集合中各元素与 R 中的某个元素均方差十分有用,可将该式展开:

$$\frac{1}{|X|} \sum_{x \in X} (x - \bar{x})^2 = \text{Var}(X) + (x' - \bar{x})^2 \quad (4-6)$$

式(4-6)之所以成立,是因为由式(4-5)可知, $\frac{1}{|x|} \sum_{x \in X} x^2 = \text{Var}(X) + \bar{x}^2$ 。

另外一条有用的性质是 X 中任意两个元素的均方差是方差的 2 倍:

$$\frac{1}{|x|^2} \sum_{x' \in X} \sum_{x \in X} (x - \bar{x})^2 = \text{Var}(X) + \frac{1}{|X|} \sum_{x' \in X} (x' - \bar{x})^2 = 2\text{Var}(X) \quad (4-7)$$

若 $X \subseteq \mathcal{R}^d$ 为一个 d 维向量构成的集合,则可为每一维定义一个方差 Var_i

(X)。可将各维方差之和 $\sum_{i=1}^d \text{Var}_i(X)$ 解释为 X 中的向量到均值向量 $\bar{x} = \frac{1}{|x|} \sum_{x \in X} x$ 的均方欧氏距离。

2. 聚类树

前面所考虑的那种简单的回归树也给出了学习聚类树的一种思路。可能这会让人觉得有些惊奇,因为回归是一种有监督学习问题,而聚类是无监督的。核心思想是回归树能够找到那些使目标值紧密分布在其均值附近的实例空间区隔——实际上,一组目标变量的方差正是它们离均值的(一元)均方欧氏距离。该方法的一种直接推广方案是使用向量型的目标变量,因为这并不会改变该方法的数学本质。更一般地,我们可引入抽象函数: $\bar{x} = \frac{1}{|x|} \sum_{x \in X} x$ 以度量任意两个实例 $x, x' \in X$ 之间的距离或不相似度(dissimilarity),这样 $\text{Dis}(x, x')$ 的值越大, x 和 x' 越不相似。一个实例集 D 的簇不相似度(cluster dissimilarity)可按下式计算:

$$\text{Dis}(D) = \frac{1}{|D|^2} \sum_{x \in D} \sum_{x' \in D} \text{Dis}(x, x') \quad (4-8)$$

在某个分裂的所有子节点上的簇不相似度的加权平均为分裂不相似度(split dissimilarity),该指标可为防 GrowTree 算法(算法 4.1)中的 BestSplit(D, F)函数所用。

关于回归树的绝大多数经验也适用于聚类树:若簇的规模较小,通常其不相似度也较低,因此很容易产生过拟合。专门保留一个剪枝集,当某些位置较低的分裂无法提升在剪枝集上的聚类一致性时,将其裁剪掉是一种推荐的方案。单样本可起主导作用,因此,若有某个分裂将第一条交易自身置为一簇时,该分裂便很难被裁剪。

那么,应如何对聚类树的叶节点进行标记?直观上,将每个簇标记为其中最具代表性的实例是一种合理的方案。若某个实例到其他所有实例的不相似度最小,则可将该实例定义为最具代表性的。然而,最有代表性的样本可能不总是唯一的。当不相似度为由数值型特征导出的欧氏距离时,既可简化确定最优分裂的计算,又可提供唯一的簇标签。实际上,均方欧氏距离就等于每个单位特征的方差之和。

以下介绍决策树算法的一个应用实例,该算法虽然简单,但是在很多场景能取得非常好的效果,值得读者一试。

决策树算法的核心是通过对数据的学习,选定判断节点,构造一颗合适的决策树。在使用决策树模型进行预测时,根据输入参数依次在各个判断节点进

行判断游走,最后到叶子节点即为预测结果。

假设我们从用户行为日志中整理出如下数据,如图 4-3 所示。我们的目的是要利用这些数据,训练决策树模型,模型训练好后,我们就可以通过任意给定的用户来源网站、位置、是否阅读过 FAQ、浏览网页数信息,预测该用户是否会进行付费以及付费类型,供运营使用。

来源网站	位置	阅读过FAQ	浏览网页数	付费类型
百度	西安	yes	18	None
谷歌	上海	no	23	Premium
搜狗	浙江	yes	24	Basic
百度	浙江	no	11	None
谷歌	西安	no	18	Basic
搜狗	上海	yes	22	None
百度	西安	no	12	None
谷歌	浙江	no	19	Basic
搜狗	西安	no	20	None
谷歌	西安	yes	16	None

图 4-3 用户日志数据

第一步,选择合适的拆分条件。

我们知道决策树是由一个个判断节点组成,每经过一个判断节点数据就会被拆分一次。上面数据中有 4 种属性,每种属性下面有多种值,我们可以按位置是否来自「浙江」进行拆分,拆分结果如下图。

来自浙江	其他地方
Basic	None
None	Premium
Basic	Basic
	None
	None
	None
	None

图 4-4 数据拆分结果

我们「拍脑袋」进行了一次拆分,到底这么拆分合不合适,是不是最佳,我们需要量化指标来进行评价,在决策树算法中,我们通过基尼不纯度或者熵来对一个集合进行的有序程度进行量化,然后引入信息增益概念对一次拆分进行量化评价。下面依次介绍。

(1) 基尼不纯度

基尼不纯度是指将来自集合中的某种结果随机应用于集合中某一数据项的预期误差率。如果集合中的每一个数据项都属于同一分类,那么推测的结果总是正确的,因此误差率是 0;如果有 4 种可能的结果均匀分布在集合内,出错可能性是 75%,基尼不纯度为 0.75。该值越高,说明拆分得越不理想,如果该值为 0,说明完美拆分。

(2) 熵

熵用来表示集合的无序程度,熵越大表示集合越混乱,反之则表示集合越有序。熵的计算公式为 $P = -P \times \log_2 P$ 。

(3) 基尼不纯度与熵对比

两者主要区别在于,熵到达峰值的过程相对慢一些。因此熵对混乱集合的「判罚」往往更重一些。通常情况下,熵的使用更加频繁。

(4) 信息增益

假设集合 U , 一次拆分后变为了两个集合 u_1 和 u_2 , 则有:

$$\text{信息增益} = E(U) - (P_{u_1} \times E(u_1) + P_{u_2} \times E(u_2))$$

E 可以是基尼不纯度或熵。

使用 P_{u_1} 和 P_{u_2} 是为了得到拆分后两个集合基尼不纯度或熵的加权平均,其中:

$$P_{u_1} = \text{Size}(u_1) / \text{Size}(U)$$

$$P_{u_2} = \text{Size}(u_2) / \text{Size}(U)$$

信息增益越大,说明整个集合从无序到有序的速度越快,本次拆分越有效。
第二步,构造决策树。

我们已经可以通过信息增益量化一次拆分的结果好坏,下一步就是构造决策树,主要步骤如下:

(1) 遍历每个决策条件(如:位置、来源网站),对结果集进行拆分;

(2) 计算该决策条件下, 所有可能的拆分情况的信息增益, 信息增益最大的拆分为本次最优拆分。

递归执行①②两步, 直至信息增益 ≤ 0 。

执行完上述步骤后, 就构造出了一颗决策树, 如图 4-5。

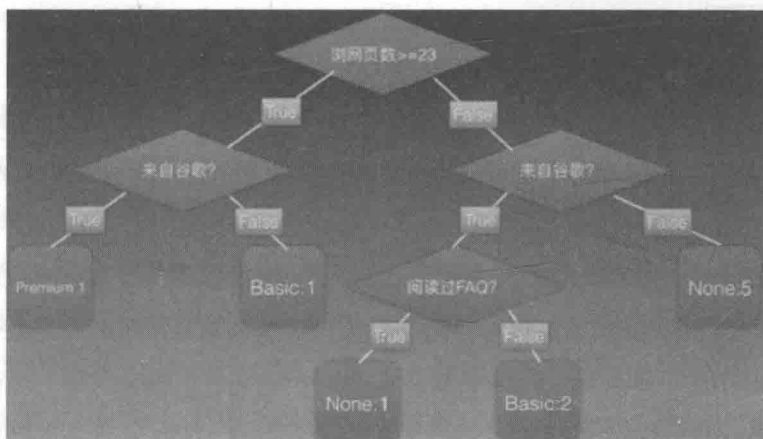


图 4-5 构造的决策树

第三步, 决策树剪枝。

(1) 为什么要剪枝

训练得出决策树存在过度拟合现象——决策树过于针对训练的数据, 专门针对训练集创建出来的分支, 其熵值可能会比真实情况有所降低。

(2) 如何剪枝

人工设置一个信息增益的阈值, 自下而上遍历决策树, 将信息增益低于该阈值的拆分进行合并。

(3) 处理缺失数据

决策树模型还有一个很大的优势, 就是可以容忍缺失数据。如果决策树中某个条件缺失, 可以按一定的权重分配继续往以后的分支走, 最终的结果可能有多个, 每个结果又一定的概率, 即

最终结果 = 某个分支的结果 $\times$ 该分支的权重 (该分支下的结果数 / 总结果数)

(4) 处理数值型数据

决策树主要解决分类问题(结果是离散数据),如果结果是数字,不会考虑这样的事实:有些数字相差很近,有些数字相差很远。为了解决这个问题,可以用方差来代替熵或基尼不纯度。

另外,从决策树发展出了更为高级复杂的随机森林,如果有兴趣读者可以去深入了解。

4.2 概率模型^①

前面已经提到,概率可用于表达模型对于某个给定实例所属类别的预期。例如,概率估计树为树模型的每个叶节点都附加了一个类概率分布,而每个沿着某条路径过滤到树模型的某个叶节点的实例,都可依据该叶节点对应的类分布进行标记。类似地,经过标定的线性模型可将实例到决策面的距离转换为类概率。这些例子中的模型都属于鉴别式(discriminative)概率模型。这类模型描述的是后验概率分布 $P(Y|X)$,其中 Y 为目标变量,而 X 为特征。即给定 X 时,这类模型可返回 Y 的概率分布。

另一大类概率模型是产生式(generative)模型。它们描述的是目标变量 Y 和特征向量 X 之间的联合概率分布 $P(Y, X)$,一旦获得了该联合分布,便可导出包含相同变量的任意条件分布或边缘分布。实际上,由于 $P(Y|X) = \sum_y P(Y=y, X)$,后验概率可表示为

$$P(Y|X) = \frac{P(Y, X)}{\sum_y P(Y=y, X)} \quad (4-9)$$

另外,由于 $P(Y, X) = P(X|Y)P(Y)$ 及目标或先验分布(通常简称为“先验”)可通过估计或假设轻松获得,因此产生式模型也可通过似然函数 $P(Y, X)$ 来描述。这类模型之所以被称为“产生式模型”,是因为我们可从上述联合概率分布中通过采样获得带有标签的新数据。此外,我们也可依据 $P(Y)$ 对类别采样,并依据 $P(Y, X)$ 对类的实例进行采样。相比之下,鉴别式模型(如概率估计树或线性分类器)描述的是 $P(Y, X)$ 而非 $P(X)$,因此可用于标记数据,但无法用于描述数据的产生机制。

^①PETER F. 机器学习[M]. 北京:人民邮电大学出版社,2016.

由于产生式模型可实现鉴别式模型所能实现的任何功能,因此似乎前者更具吸引力。然而,产生式模型也有许多难以克服的缺陷。首先,存储联合概率分布所需的空间与特征维数呈指数关系。这就要求必须通过简化一些假设(如特征之间的独立性),而一旦这种假设不符合特定问题域的特点,势必会降低结果的精确性。人们对产生式模型最常见的批判是,这种方法对建模的精确性是以牺牲 $P(Y, X)$ 建模的精确性为代价的。然而,人们对于这个问题至今尚未完全理解。在某些情形下,对于 $P(X)$ 的了解还会有助于人们理解给定问题域。例如,如果依据某些实例出现的概率很小,则我们就可以不那么在意它们被误分类。

概率视角最吸引人的特征之一是,我们因此可将学习视为一个减少不确定性的过程。例如,一个服从均匀分布的类先验告诉我们,在了解待分类实例之前,我们完全不确定应将该实例划分到哪个类别。如果在观测到实例之后所获得的后验概率的均匀性有所降低,则意味着我们已将分类的不确定性减少。每次接收到新息(即新的信息)之后,我们可重复该过程,即将前一步所获得的后验概率作为后一步的先验。理论上,该过程可应用于任何未知量的估计。

4.2.1 正态分布及其几何意义

如果仅考虑定义在欧氏空间上的概率分布,则我们可建立概率模型与几何模型之间的联系。欧氏空间中最常见的分布莫过于正态分布(normal distribution),亦称高斯分布(Gaussian);这里与一元正态分布和多元正态分布有关的最重要的知识点请参加相关书籍。接下来,我们首先讨论两类情形下的一元高斯分布。假设实数服从某个混合模型(mixture model),即每个类别都拥有特定的概率分布(称为该混合模型的一个分量,component)。这里我们假设该混合模型为高斯混合模型(Gaussian Mixture Model, GMM),即该混合模型的两个分量均为高斯分布。因此,

$$P(x|\oplus) = \frac{1}{\sqrt{2\pi}\sigma^{\oplus}} \exp\left(-\frac{1}{2}\left[\frac{x-\mu^{\oplus}}{\sigma^{\oplus}}\right]^2\right)$$

$$P(x|\ominus) = \frac{1}{\sqrt{2\pi}\sigma^{\ominus}} \exp\left(-\frac{1}{2}\left[\frac{x-\mu^{\ominus}}{\sigma^{\ominus}}\right]^2\right)$$

其中, $\mu^{\oplus}$ 和 $\sigma^{\oplus}$ 分别为正类的均值和标准差,而 $\mu^{\ominus}$ 和 $\sigma^{\ominus}$ 分别为负类的均值和

标准差。由此可导出如下的似然比：

$$LR(x) = \frac{p(x|\oplus)}{p(x|\ominus)} = \frac{\sigma^\ominus}{\sigma^\oplus} \exp\left(-\frac{1}{2}\left[\left(\frac{x-\mu^\oplus}{\sigma^\oplus}\right)^2\right] - \left(\frac{x-\mu^\ominus}{\sigma^\ominus}\right)^2\right) \quad (4-10)$$

我们首先来考虑混合模型中的两个分量具有相同标准差的情形，即据此可将式(4-10)中的指数项简化为

$$-\frac{1}{2\sigma^2}[(x-\mu^\oplus)^2 - (x-\mu^\ominus)^2] = -\frac{1}{2\sigma^2}[x^2 - 2\mu^\oplus x + \mu^{\oplus 2} - (x^2 - 2\mu^\ominus x + \mu^{\ominus 2})]$$

因此，似然比可表示为 $LR(x) = \exp(\gamma(x-\mu))$ ，其中 $\gamma = (\mu^\oplus - \mu^\ominus)/\sigma^2$ 为正类和负类均值之差与方差之比，而 $\mu = (\mu^\oplus + \mu^\ominus)/2$ 为两类均值的中心点。因此最大似然决策阈值(当 x 取该值时， $LR(x)=1$)为 $x_{ML} = \mu$ 。

若 $\sigma^\oplus \neq \sigma^\ominus$ ，则式(4-10)中的 x^2 项无法消去。此时会产生两个决策面，且有一个类别的决策域是非连续的。

高维空间中有可能出现非连续的决策域。下面给出一个对应于 $m=2$ 的例子。

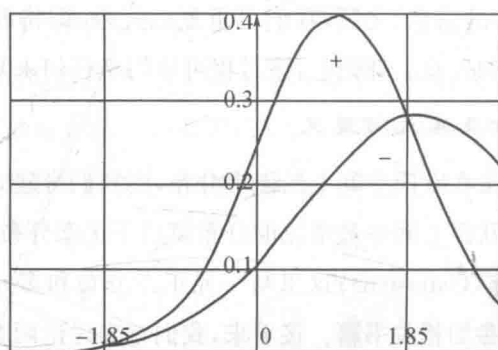


图 4-6 正样本来自均值和标准差均为 1 的高斯分布，而负样本均来自均值和标准差为 2 的高斯分布

图 4-6 中，两个分布在 $x = \pm 1.85$ 处相交，这意味着正例的最大似然区域为闭区间 $[-1.85, 1.85]$ ，因此负例的最大似然区域是不连续的

4.2.2 属性数据的概率模型

属性变量或特征(也称为离散变量或名义变量)在机器学习中俯拾即是。伯努利分布最常见的形式可能是描述某个单词是否在某篇文档中出现。即，对于词汇表中的第 i 个单词，存在一个与之对应的、由某个伯努利分布控制的

随机变量 X_i 。在位向量(bit vector) $X=(X_1, \dots, X_k)$ 上的联合分布被称为多元伯努利分布。具有两个以上结果的变量也很常见。例如,电子邮件中每个单词的位置对应于一个拥有 k 个结果的属性变量,其中 k 为词汇表的长度。多项分布实际上是一个计数向量,即一个记录了词汇表中所有单词在文档中的出现次数的直方图。这便提供了另外一种对文本文档建模的方法,它允许单词出现的频数对文档的分类结果产生影响。

这些文档模型都较为常用。尽管它们存在一些差异,但都假设不同单词是否出现是相互独立的,该假设通常被称为朴素贝叶斯假设(naive Bayes assumption)。在多项文档模型中,这种假设是由多项分布确定的,即不同位置出现的单词是依据同一个属性分布独立抽取得到的。在多元伯努利模型中,我们假设位向量中的各个位是统计独立的,于是可将某个位向量 $(X_1, \dots, X_k)$ 的联合概率分解为各分量的概率的成绩。实际中,这样的单词独立性假设往往是不成立的:如果我们已经获悉某封电子邮件中包含了单词“Viagra”,则可以确定该邮件中也包含单词“pill”。无论在何种情况下,我们有这样一条经验,即虽然依据朴素贝叶斯假设所得到的概率估计量的质量较差,但它往往不会影响排序性能。这意味着,若我们精心选择分类阈值,则通常也可以获得好的分类性能。

1. 利用朴素贝叶斯模型实现分类

假设我们已为数据 X 选择了一种可能的分布作为模型。在分类中,我们进一步假设该分布依赖于具体的类别,因而 $P(X|X=spam)$ 和 $P(X|X=ham)$ 为两个不同的分布。这两个分布的差异越明显,特征 X 在分类中能发挥的作用就越大。因此,对于某封电子邮件,我们可同时计算 $P(X=a;|y=pam)$ 和 $P(X=a;|Y=ham)$,并运用下列决策规则之一:

最大似然(ML) $\operatorname{argmax}_y P(X=x|Y=y)$

最大后验概率(MAP) $\operatorname{argmax}_y P(X=x|Y=y)P(Y=y)$

重标定似然 $\operatorname{argmax}_y w_y P(X=x|Y=y)$

前两个决策规则之间的关系是:ML 分类等价于均匀分类分布下的 MAP 分类。第三个决策规则对前两个进行了推广,因为它将类分布替换为从数据中学习到的的一组权值,这就使得纠正似然中的估计误差成为可能,稍后我们将了解到这一点。

2. 训练朴素贝叶斯模型

训练概率模型时,通常要估计该模型所使用的分布的参数。伯努利分布的参数可通过统计 n 次试验中成功的次数 d 并取来估计。换言之,我们需要为每个类别统计有多少封电子邮件中包含当前关注的单词。这种相对频率估计量通常需要引入伪计数项来进行平滑。

表示依据某些固定分布得到的虚拟试验结果。对于伯努利分布,最常见的平滑操作是拉普拉斯修正,其中涉及两个虚拟试验,一个获得成功,一个失败。因此,相对频率估计量就被修正为 $(d+1)/(n+2)$ 。从贝叶斯角度看,这相当于采用了一个均匀分布的先验来表示我们的初始信念——成功和失败是等可能的。在适当的情况下,我们可通过引入更多的虚拟试验来增强先验的影响,这意味着要想使估计量偏离先验,必须借助更多的数据。对于属性分布,平滑处理为个类别中的每一个增加了一个伪计数项,从而有经过平滑的估计量 $(d+1)/(n+k)$ 。 m 估计量(m -estimate) 通过将伪计数的总数 m 和它们分布在不同类别中的方式参数化对此进行了推广。第 i 个类别的估计量被定义为 $(d+pim)/(n+m)$,其中 p_i 是一个关于类别的分布,即 $\sum_{i=1}^k p_i = 1$ 。请注意,经过平滑处理的相对频率估计量,以及这些估计量的乘积,永远不会取到极值 $\hat{\theta}=0$ 或 $\hat{\theta}=1$ 。

总之,朴素贝叶斯模型是处理文本数据、属性数据及属性和实值型混合的数据的利器。因为排序性能优良,所以它作为概率模型(未正确标定的概率估计)所存在的主要缺陷可以忽略不计。关于朴素贝叶斯模型的另一个悖论是它算不上是一个贝叶斯模型。一方面,我们已经了解到这个性能不佳的概率估计需要使用重加权的似然,这避免了使用贝叶斯公式;另一方面,在训练朴素贝叶斯模型时,我们使用了最大似然参数估计,而真正的贝叶斯方法并不会只关注某个参数取值,而是运用整个后验分布。

4.2.3 通过优化条件似然实现鉴别式学习

本小节简要介绍最常见的一种鉴别式模型——logistic 回归模型。理解 logistic 回归的最简单方式是将其视为一个线性分类器。但需要注意,这仍与 logistic 回归模型存在显著差异:在 logistic 模型中,标定是训练算法的一部分,

而非后处理步骤。在产生式模型中,决策面是对每类的分布建模后的副产品,而在 logistic 回归模型则直接对决策面建模。例如,若各类之间存在交叠,则 logistic 回归模型会将决策面定在各类具有最大重叠之处,而并不具体考虑每类的样本所构成的“形状”如何。因此,该模型所形成的决策面与产生式分类器的决策面存在明显的区别(如图 4-7 所示)。

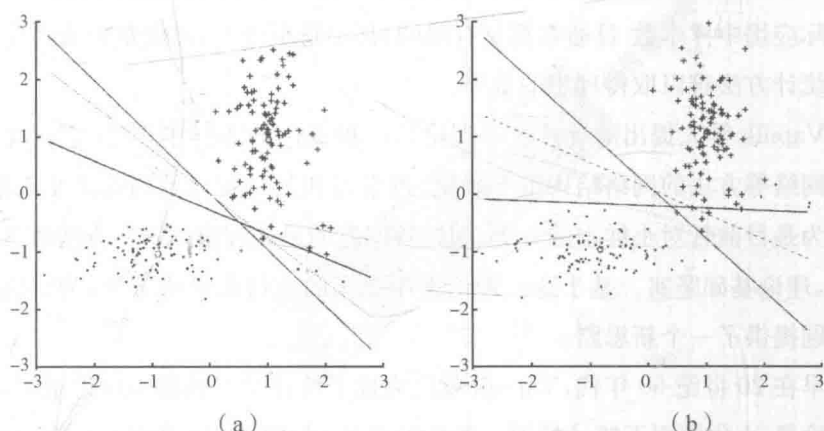


图 4-7 (a)logistic 回归模型的性能超过了基本线性分类器以及最小二乘分类器;(b)logistic 回归模型的性能逊于另外两种方法

图 4-7(a)中,因为后二者对于各类样本构成的形状更加敏感,而 logistic 回归关注各类样本交叠之处。图 4-7(b)中,该图所展示的数据集与(a)略有不同。在该数据集上,logistic 回归模型的性能逊于另外两种方法,因为它过于关注追踪从正例密度最大的位置向负例密度最大的位置的过渡

4.3 支持向量机

支持向量机(Support Vector Machines, SVM)方法是建立在统计学习模型理论的 VC 维理论和结构风险最小原理基础上的,根据有限的样本信息在模型的复杂性(即对特定训练样本的学习精度)和学习能力(即无错误地识别任意样本的能力)之间寻求最佳折中,以求获得最好的推广能力。其分类的过程就是一个机器学习的过程。

4.3.1 SVM 的基本思想

学习的本质是归纳,对此经典逻辑很少办法。模式识别、回归分析、密度估

计是学习的三个基本内容。学者们应用概率统计和函数逼近等方法取得了很多研究成果,特别是在处理线性问题上,有些甚至是完满的。但对于本质上非线性的问题还缺少较好的结果。

当我们面对数据而又缺乏理论模型时,统计分析方法是最先采用的方法。然而传统的统计方法只有在样本数量趋于无穷大时才能有理论上的保证。而在实际应用中样本数目通常都是有限的,甚至是小样本,对此基于大数定律的传统统计方法难以取得理想的效果。

Vapnik 等人提出的统计学习理论是一种专门的小样本理论,它避免了人工神经网络等方法的网络结构难于确定、过学习和欠学习以及局部极小等问题,被认为是目前针对小样本的分类、回归等问题的最佳理论。这一方法数学推导严密,理论基础坚实。基于这一理论近年提出的支持向量机方法,为解决非线性问题提供了一个新思路。

早在 20 世纪 60 年代,Vapnik 就已完成了统计学习的基本理论结果,如经验风险最小化原则下统计学习一致性的条件(收敛性、收敛的可控性、收敛与概率测度定义的无关性,号称机器学习理论的“三个里程碑”)、关于统计学习方法推广能力的界的理论以及在此基础上建立的小样本归纳推理原则等。直到 20 世纪 90 年代中期,实现统计学习理论和原则的实用化算法——SVM 方法才被逐渐完整提出,并在模式识别等人工智能领域成功应用,受到广泛关注。

1. 最优超平面的概念

考虑 P 个线性可分样本 $\{(X^1, d^1), (X^2, d^2), \dots, (X^p, d^p), \dots, (X^P, d^P)\}$, 对于任一输入样本 X^p , 其期望输出为 $d^p = \pm 1$, 分别代表两类的类别标识。用于分类的超平面方程为

$$W^T X + b = 0 \quad (4-11)$$

式中, X 为输入向量, W 为权值向量, b 为偏置, 相当于上一章中的负阈值 ($b = -T$), 则有

$$W^T X^p + b > 0, \text{ 当 } d^p = +1$$

$$W^T X^p + b < 0, \text{ 当 } d^p = -1$$

由式(4-1)定义的超平面与最近的样本点之间的间隔称为分离边缘, 用 ρ 表示。支持向量机的目标是找到一个分离边缘最大的超平面, 即最优超平面。

图4-8给出二维平面中最优超平面的示意图。可以看出,最优超平面能提供两类之间最大可能的分离,因此确定最优超平面的权值 W_0 和偏置 b_0 应是唯一的。在式(4-11)定义的一簇超平面中,最优超平面的方程应为

$$W^T X_0 + b_0 = 0 \quad (4-12)$$

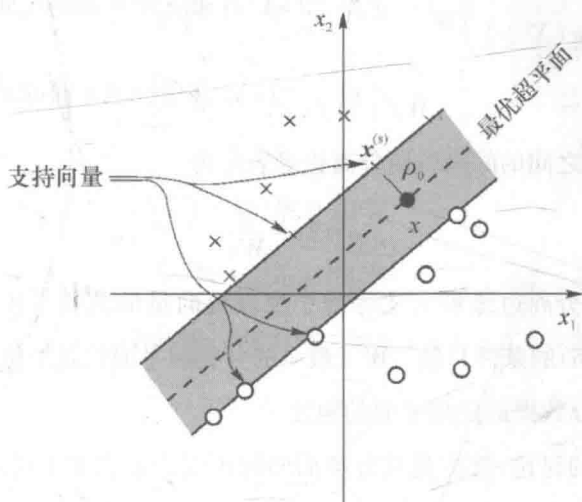


图4-8 二维平面中的最优超平面

由解析几何知识可得样本空间任一点到最优超平面的距离为

$$r = \frac{W_0^T X + b_0}{W_0} \quad (4-13)$$

从而有判别函数

$$g(X) = rW_0 = W_0^T X + b_0 \quad (4-14)$$

给出从 X 到最优超平面的距离的一种代数度量。

将判别函数进行归一化,使所有样本都满足

$$\begin{cases} W^T X^p + b \geq 1, \text{当 } d^p = +1 \\ W^T X^p + b \leq -1, \text{当 } d^p = -1 \\ p = 1, 2, \dots, P \end{cases} \quad (4-15)$$

则对于离最优超平面最近的特殊样本 X_s 满足 $g(X_s) = 1$, 称为支持向量。由于支持向量最靠近分类决策面,是最难分类的数据点,因此这些向量在支持向量机的运行中起着主导作用。

式(4-15)中的两行也可以组合起来用下式表示

$$d^p(W^T X^p + b) \geq 1, p=1, 2, \dots, P \quad (4-16)$$

其中, W_0 用 W 代替。

由式(4-13)可导出从支持向量到最优超平面的代数距离为

$$r = \frac{g(X^s)}{W_0} = \begin{cases} \frac{1}{W_0} & d^s = +1, X^s \text{ 在最优超平面的正面} \\ -\frac{1}{W_0} & d^s = -1, X^s \text{ 在最优超平面的负面} \end{cases} \quad (4-17)$$

因此, 两类之间的间隔可用分离边缘表示为

$$\rho = 2y = \frac{2}{W_0} \quad (4-18)$$

上式表明, 分离边缘最大化等价于使权值向量的范数 $\|W\|$ 最小化。因此, 满足式(4-16)的条件且使 $\|W\|$ 最小的分类超平面就是最优超平面。

2. 线性可分数据最优超平面的构建

根据上面的讨论, 建立最优分类面问题可以表示成如下的约束优化问题, 即对于给定的训练样本 $\{(X^1, d^1), (X^2, d^2), \dots, (X^p, d^p), \dots, (X^P, d^P)\}$, 找到权值向量 W 和阈值 T 的最优值, 使其在式(4-16)的约束下, 最小化代价函数

$$\Phi(W) = \frac{1}{2} W^T W = \frac{1}{2} W^T W \quad (4-19)$$

这个约束优化问题的代价函数是 W 的凸函数, 且关于 W 的约束条件是线性的, 因此可以用 Lagrange 系数方法解决约束最优问题。引入 Lagrange 函数如下

$$L(W, b, \alpha) = \frac{1}{2} W^T W - \sum_{p=1}^P \alpha_p [d^p (W^T X^p + b) - 1] \quad (4-20)$$

式中, $\alpha > 0, p=1, 2, \dots, P$ 称为 Lagrange 系数。式(4-10)中的第一项为代价函数 $\Phi(W)$, 第二项非负, 因此最小化 $\Phi(W)$ 就转化为求 Lagrange 函数的最小值。观察 Lagrange 函数可以看出, 欲使该函数值最小化, 应使第一项 $\Phi(W) \downarrow$, 使第二项 $\uparrow$ 。为使第一项最小化, 将式(4-11)对 W 和 b 求偏导, 并使结果为零

$$\frac{\partial L(W, b, a)}{\partial W} = 0 \quad (4-21)$$

$$\frac{\partial L(W, b, a)}{\partial b} = 0$$

利用式(4-10)和式(4-11),经过整理可导出最优化条件 1

$$W = \sum_{p=1}^P \alpha_p d^p X^p \quad (4-22)$$

利用式(4-10)和式(4-11)可导出最优化条件 2

$$\sum_{p=1}^P \alpha_p d^p = 0 \quad (4-23)$$

为使第二项最大化,将式(4-10)展开如下

$$L(W, b, a) = \frac{1}{2} W^T W - \sum_{p=1}^P \alpha_p d^p W^T W^p - b \sum_{p=1}^P \alpha_p d^p + \sum_{p=1}^P \alpha_p$$

根据式(4-13),上式中的第三项为零。根据式(4-12),可将上式表示为

$$\begin{aligned} L(W, b, a) &= \frac{1}{2} W^T W - W^T \sum_{p=1}^P \alpha_p d^p X^p + \sum_{p=1}^P \alpha_p \\ &= \frac{1}{2} W^T W - W^T W + \sum_{p=1}^P \alpha_p \\ &= \frac{1}{2} W^T W + \sum_{p=1}^P \alpha_p \end{aligned}$$

根据式(4-22)可得到

$$W^T W = W^T \sum_{p=1}^P \alpha_p d^p X^p = \sum_{p=1}^P \sum_{j=1}^P \alpha_p \alpha_j d^p d^j (X^p)^T X^p \quad (4-24)$$

至此,原来的最小化 $L(W, b, a)$ 函数问题转化为一个最大化函数 $Q()$ 的“对偶”问题,即:给定训练样本 $\{(X^1, d^1), (X^2, d^2), \dots, (X^P, d^P), \dots, (X^P, d^P)\}$,求解使式(4-24)为最大值的 Lagrange 系数 $\{\alpha_1, \alpha_2, \dots, \alpha_P, \alpha_P\}$,并满足约束条件 $\sum_{p=1}^P \alpha_p d^p = 0; \alpha_p \geq 0, p=1, 2, \dots, P$ 。

以上为不等式约束的二次函数极值问题(Quadratic Programming, QP)。由 Kuhn 唱 Tucker 定理知,式(4-14)的最优解必须满足以下最优化条件(KKT 条件)

$$\alpha_p [(W^T X^p + b) d^p - 1] = 0, p=1, 2, \dots, P \quad (4-25)$$

可以看出,在两种情况下上式中的等号成立:一种情况是 α_p 为零;另一种情况是 α_p 不为零而 $(W^T X^p + b) d^p = 1$ 。显然,第二种情况仅对应于样本为支持向量的情况。

设 $Q(\alpha)$ 的最优解为 $\{\alpha_1, \alpha_2, \dots, \alpha_p, \alpha_P\}$, 可通过式(4-22)计算最优权值向量, 其中多数样本的 Lagrange 系数为零, 因此

$$W_0 = \sum_{p=1}^P \alpha_p d^p X^p = \sum_{\text{所有向量}} \alpha_p d^s X^s \quad (4-26)$$

即最优超平面的权向量是训练样本向量的线性组合, 且只有支持向量影响最终的划分结果, 这就意味着如果去掉其他训练样本再重新训练, 得到的分类超平面是相同的。但如果一个支持向量未能包含在训练集内时, 最优超平面会被改变。

利用计算出的最优权值向量和一个正的支持向量, 可通过式(4-15)进一步计算出最优偏置

$$d_0 = 1 - W_0^T X^s \quad (4-27)$$

求解线性可分问题得到的最优分类判别函数为

$$f(X) = \text{sgn} \left[\sum_{p=1}^P \alpha_p d^p (X^p)^T + b_0 \right] \quad (4-28)$$

在上式中的 P 个输入向量中, 只有若干个支持向量的 Lagrange 系数不为零, 因此计算复杂度取决于支持向量的个数。

对于线性可分数据, 该判别函数对训练样本的分类误差为零, 而对非训练样本具有最佳泛化性能。

3. 非线性可分数据最优超平面的构建

若将上述思想用于非线性可分模式的分类时, 会有一些样本不能满足式(4-16)的约束, 而出现分类误差。因此需要适当放宽该式的约束, 将其变为

$$d^p (W^T X^p + b) \geq 1 - \xi_p, p=1, 2, \dots, P \quad (4-29)$$

式中引入了松弛变量 $\xi_p \geq 0, p=1, 2, \dots, P$, 它们用于度量一个数据点对线性可分理想条件的偏离程度。当 $0 \leq \xi_p \leq 1$ 时, 数据点落入分离区域的内部, 且在分类超平面的正确一侧; 当 $\xi_p > 1$ 时, 数据点进入分类超平面的错误一侧; 当 $\xi_p = 0$ 时, 相应的数据点即为精确满足式(4-16)的支持向量 X^s 。

建立非线性可分数据的最优超平面可以采用与线性可分情况类似的方法, 即对于给定的训练样本 $\{(X^1, d^1), (X^2, d^2), \dots, (X^p, d^p), \dots, (X^P, d^P)\}$, 寻找权值 W 和阈值 T 的最优值, 使其在式(4-29)的约束下, 最小化关于权值 W 和松弛变量的代价函数

$$\Phi(W, \xi) = \frac{1}{2} W^T W + C \sum_{p=1}^P \xi_p \quad (4-30)$$

式中, C 是使用者选定的正参数。

与上一节采用的方法类似, 利用 Lagrange 系数方法解决约束最优问题。需要注意的是, 在引入 Lagrange 函数时, 使式(4-20)中的 1 被代替, 从而使 Lagrange 函数变为

$$L(W, b, a) = \frac{1}{2} W^T W - \sum_{p=1}^P \alpha_p [d^p (W^T X^p + b) - 1 + \xi_p] \quad (4-31)$$

对式(4-31)采用与上一节中类似的推导, 得到非线性可分数据的对偶问题的表示为: 给定训练样本 $\{(X^1, d^1), (X^2, d^2), \dots, (X^p, d^p), \dots, (X^P, d^P)\}$, 求解使以下目标函数

$$Q(\alpha) = \sum_{p=1}^P \alpha_p - \frac{1}{2} \sum_{p=1}^P \sum_{j=1}^P \alpha_p \alpha_j d^p d^j (X^p)^T X^j$$

为最大值的 Lagrange 系数 $\{\alpha_1, \alpha_2, \dots, \alpha_p, \alpha_P\}$, 并满足以下约束条件

$$\begin{aligned} \sum_{p=1}^P \alpha_p d^p &= 0 \\ 0 \leq \alpha_p &\leq C, p=1, 2, \dots, P \end{aligned} \quad (4-32)$$

可以看出在上述目标函数中, 松弛变量 $\xi_p, p=1, 2, \dots, P$ 和它们的 Lagrange 系数都未出现, 因此线性可分的目标函数与非线性可分的目标函数表达式完全相同。不同的只是线性可分情况下的约束条件 $\alpha_p \geq 0$ 在非线性可分情况下被替换为约束更强的 $0 \leq \alpha_p \leq C$, 因此线性可分情况下的约束条件 $\alpha_p \geq 0$ 可以看作非线性可分情况下的一种特例。

此外, W 和 b 的最优解必须满足的 KuhnTucke 最优化条件改变为

$$\alpha_p [(W^T X^p + b) d^p - 1 + \xi_p] = 0, p=1, 2, \dots, P \quad (4-33)$$

最终推导得到的 W 和 b 的最优解计算式以及最优分类判别函数与式(4-26)、式(4-27)和式(4-28)完全相同。

4.3.2 非线性支持向量机

在解决模式识别问题时, 经常遇到非线性可分模式的情况。支持向量机的方法是, 将输入向量映射到一个高维特征向量空间, 如果选用的映射函数适当且特征空间的维数足够高, 则大多数非线性可分模式在特征空间中可以转化为

线性可分模式,因此可以在该特征空间构造最优超平面进行模式分类,这个构造与内积核相关。

1. 基于内积核的最优超平面

设 X 为 N 维输入空间的向量,令 $\Phi(X) = [\varphi_1(X), \varphi_2(X), \dots, \varphi_M(X)]^T$ 表示从输入空间到 M 维特征空间的非线性变换,称为输入向量 X 在特征空间诱导出的“像”。参照前述思路,可以在该特征空间定义构建一个分类超平面

$$\sum_{j=1}^M \omega_j \varphi_j(X) + b = 0 \quad (4-34)$$

式中的 $\omega_j, j=1, 2, \dots, M$ 为将特征空间连接到输出空间的权值, b 为偏置或阈值。 $\varphi_0(X) = 1, \omega_0 = b$ 上式可简化为

$$\sum_{j=0}^M \omega_j \Phi_j(X) = 0 \quad (4-35)$$

或写成

$$W^T \Phi(X) = 0 \quad (4-36)$$

将适合线性可分模式输入空间的式(4-12)用于特征空间中线性可分的“像”,只需用 $\Phi(X)$ 替换 X ,得到

$$W = \sum_{p=1}^p \alpha_p \Phi(X)^p \quad (4-37)$$

将上式代入式(4-26)可得特征空间的分类超平面为

$$\sum_{p=1}^p \alpha_p d^p \Phi(X) \Phi^T X^p = 0 \quad (4-38)$$

式中, $\Phi(X) \Phi^T X^p$ 表示第 p 个输入模式在特征空间的像 $\Phi(X^p)$ 与输入向量 X 在特征空间的像 $\Phi(X)$ 的内积,因此在特征空间构造最优超平面时,仅使用特征空间中的内积。若能找到一个函数 $K(\cdot)$,使得

$$K(X, X^p) = \Phi(X) \Phi^T X^p = \sum_{j=1}^M \Phi_j^T X^p, p=1, 2, \dots, p \quad (4-39)$$

则在特征空间建立超平面时无须考虑变换的形式。称为内积核函数。泛函分析中的 Mercer 定理给出作为核函数的条件: $K(X, X')$ 表示一个连续的对称核,其中 X 定义在闭区间 $a \leq X \leq b, X'$ 类似。核函数 $K(X, X')$ 可以展开为级数

$$K(X, X') = \sum_{i=1}^{\infty} \lambda_i \Phi_i(X') \quad (4-40)$$

式中所有数据均大于 0。保证式(4-30)一致收敛的充要条件是

$$\int_b^a \int_b^a K(X, X') \Phi(X) \Phi(X') dX dX' \geq 0 \quad (4-41)$$

对于所有满足 $\int_b^a \Phi^2(X) dX < \infty$ 的 $\Phi(\cdot)$ 成立。

可以看出式(4-39)对于内积核函数 $K(X, X^p)$ 的展开是 Mercer 定理的一种特殊情况。Mercer 定理指出如何确定一个候选核是不是某个空间的内积核,但没有指出如何构造函数 $\Phi_i(X)$ 。

对核函数 $K(X, X^p)$ 的要求是满足 Mercer 定理,因此其选择有一定的自由度。下面给出 3 种常用的核函数。

(1) 多项式核函数

$$K(X, X^p) = [(X \cdot X^p) + 1]^q \quad (4-42)$$

采用该函数的支持向量机是一个 q 阶多项式分类器,其中 q 为由用户决定的参数。

(2) Gauss 核函数

$$K(X, X^p) = \exp\left(-\frac{|X - X^p|^2}{2\sigma^2}\right) \quad (4-43)$$

采用该函数的支持向量机是一种径向集函数分类器。

(3) Sigmoid 核函数

$$K(X, X^p) = \tanh[k(X \cdot X^p)] + c \quad (4-44)$$

采用该函数的支持向量机实现的是一个单隐层感知器神经网络。

使用内积核在特征空间建立的最优超平面定义为

$$\sum_{p=1}^P \alpha_p d^p K(X, X^p) = 0 \quad (4-45)$$

2. 非线性支持向量机神经网络

支持向量机的思想是,对于非线性可分数据,在进行非线性变换后的高维特征空间实现线性分类,此时最优分类判别函数为

$$f(X) = \text{sgn}[\sum_{p=1}^P \alpha_p d^p K(X, X^p) + b_0] \quad (4-46)$$

令支持向量的数量为 N_s ,去除系数为零的项,上式可改写为

$$f(X) = \text{sgn}[\sum_{s=1}^{N_s} \alpha_s d^s K(X; X^s) + b_0] \quad (4-47)$$

从支持向量机分类判别函数的形式上看,它类似于一个 3 层前馈神经网络。其中隐层节点对应于输入样本与一个支持向量的内积核函数,而输出节点

对应于隐层输出的线性组合。图 4-9 给出支持向量机神经网络的示意图。

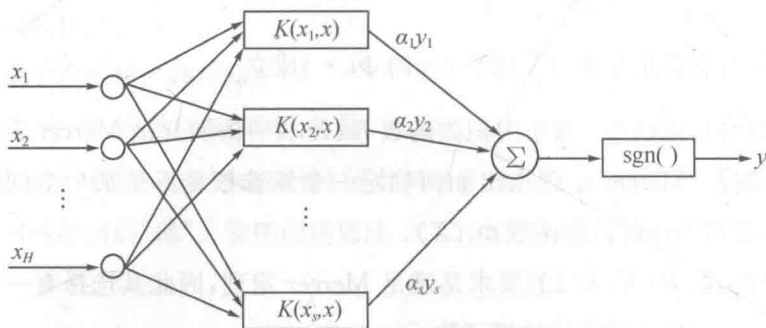


图 4-9 支持向量机神经网络

设计一个支持向量机时,只需选择满足 Mercer 条件的核函数而不必了解将输入样本变换到高维特征空间的 $\Phi(\cdot)$ 的形式,但下面给出的简单的核函数实际上能够构建非线性映射 $\Phi(\cdot)$ 。

设输入数据为二维平面的向量 $X=[x_1, x_2]^T$, 共有 3 个支持向量, 因此应将二维输入向量非线性映射为三维空间的向量 $\Phi(X)=[\varphi_1(X), \varphi_2(X), \dots, \varphi_M(X)]^T$ 。选择 $K(X_i, X_j)=[(X_i)^T \cdot X_j]/2$, 使映射 $\Phi(\cdot)$ 从 $R_2 \rightarrow R_3$ 满足

$$[(X_i)^T \cdot X_j]/2 = \Phi^T(X)\Phi(X)$$

对于给定的核函数,映射 $\Phi(\cdot)$ 和特征空间的维数都不是唯一的,例如,对于本例的情况可选 $\Phi(X)=[x_1^2, \varphi_2(X), \varphi_3(X)]^T$, 或 $\Phi(X)=[\varphi_1(X), \varphi_2(X), \varphi_3(X)]^T$ 。

4.3.3 支持向量机的学习算法

在能够选择变换 Φ (取决于设计者在这方面的知识) 的情况下,用支持向量机进行求解的学习算法如下。

(1) 通过非线性变换 Φ 将输入向量映射到高维特征空间。

(2) 在约束条件 $\sum_{p=1}^P \alpha_p d^p = 0, 0 \leq \alpha_p \leq C$ 或 $(\alpha_p \geq 0), p=1, 2, \dots, P$ 下求解使目标函数

$$Q(\alpha) = \sum_{p=1}^P \alpha_p - \frac{1}{2} \sum_{p=1}^P \sum_{j=1}^P \alpha_p \alpha_j d^p d^j \Phi^T(X^p) \Phi(X^j) \quad (4-48)$$

最大化的 α_p 。

(3) 计算最优权值

$$W_0 = \sum_{j=1}^P \alpha_j d^j \Phi(X^j) \quad (4-49)$$

(4) 对于待分类模式 X , 计算分类判别函数

$$f(X) = \text{sgn}[\sum_{p=1}^P \alpha_p d^p \Phi^T(X^p) \Phi(X) + b_0] \quad (4-50)$$

根据 $f(X)$ 为 1 或 -1, 决定 X 的类别归属。

若能选择一个内积核函数 $K(X, X^p)$, 可避免进行变换, 此时用支持向量机进行求解的学习算法如下:

(1) 准备一组训练样本 $\{(X^1, d^1), (X^2, d^2), \dots, (X^p, d^p), \dots, (X^P, d^P)\}$ 。

(2) 在约束条件 $\sum_{p=1}^P \alpha_p d^p = 0, 0 \leq \alpha_p \leq C$ 或 $(\alpha_p \geq 0), p=1, 2, \dots, P$ 下求解使目标函数

$$Q(\alpha) = \sum_{p=1}^P \alpha_p - \frac{1}{2} \sum_{p=1}^P \sum_{j=1}^P \alpha_p \alpha_j d^p d^j K(X^p, X^j) \quad (4-51)$$

最大化的 α_p , 其中 $K(X^p, X^j), p, j=1, 2, \dots, P$, 可以看作是 $P \times P$ 对称矩阵 K 的第 pj 项元素。

(3) 计算最优权值

$$W_0 = \sum_{j=1}^P \alpha_j d^j Y^j \quad (4-52)$$

Y 为隐层输出向量。

(4) 对于待分类模式 X , 计算分类判别函数

$$f(X) = \text{sgn}[\sum_{p=1}^P \alpha_p d^p K(X, X^p) + b_0] \quad (4-53)$$

根据 $f(X)$ 为 1 或 -1, 决定 X 的类别归属。

上面讨论的支持向量机只能解决二分类问题, 目前没有一个统一的方法将其推广到多分类的情况, 但已有不少设计者针对具体问题提出了值得借鉴的方法, 读者可参考相关论文。

支持向量机常被用于径向基函数网络和多层感知器的设计中。在径向基函数类型的支持向量机中, 径向基函数的数量和它们的中心分别由支持向量的个数和支持向量的值决定, 而传统的 RBF 网络对这些参数的确定则依赖于经验知识。在单隐层感知器类型的支持向量机中, 隐节点的个数和它们的权值向量分别由支持向量的个数和支持向量的值决定。

与径向基函数和多层感知器相比, 支持向量机的算法不依赖于设计者的经验知识, 且最终求得的是全局最优值而不是局部极值, 因而具有良好的泛化能

力而不会出现过学习现象。支持向量机由于算法复杂导致训练速度较慢,其中的主要原因是在算法第(2)步的寻优过程中涉及大量矩阵运算。目前提出的一些改进训练算法是基于循环迭代的思想,下面介绍 3 类改进算法的主要思路。

(1) Vapnik 等提出的块算法

对于给定的训练集,通过某种迭代方式逐步排除非支持向量。首先,选择一部分样本构成工作样本集进行训练,剔除其中的非支持向量,并用训练结果对剩余样本进行检验,将不符合训练结果的样本或其中一部分与本次结果的支持向量合并成为一个新的工作样本集,重新进行训练。反复进行直到获得最优结果。块算法适合于支持向量的数目远远小于训练样本数的情况。当支持向量的数量较多时,随着算法迭代次数的增多,工作样本集会越来越大。

(2) Qsuna 等提出的分解算法

算法的主要思想是将训练样本分为工作集和非工作集,工作集中的样本数量远小于总样本数。每次只对工作集中的样本进行训练,而固定非工作集样本。该算法的关键在于确定一种最优的工作样本集选择方法。该算法需要注意的问题是:①应用 KKT 优化条件推出问题的终止条件;②工作集训练样本的选择方法应保证分解算法快速收敛,且计算量少。

(3) Platt 的 SMO 算法

上述两种算法都需要对分解后的子系统求解 QP 问题的内循环。尽管求解的 QP 问题比原问题规模小,但仍需用数值法求解,从而带来一些计算精度和计算复杂性方面的问题。SMO(Sequential Minimal Optimization)算法的工作空间只包含 2 个样本,在每一步迭代时只对 2 个 Lagrange 系数进行优化,因此该算法中只需一段简洁的程序代码就能解决 QP 优化问题。尽管在 SMO 算法中 QP 子问题增多,但总的计算速度大为提高,使该算法成为在解决实际问题中应用最广的方法。

4.3.4 支持向量机的应用研究现状

SVM 方法在理论上具有突出的优势,贝尔实验室率先对美国邮政手写数字库识别研究方面应用了 SVM 方法,取得了较大的成功。在随后的近几年内,有关 SVM 的应用研究得到了很多领域的学者的重视,在人脸检测、验证和

识别、说话人/语音识别、文字/手写体识别、图像处理及其他应用研究等方面取得了大量的研究成果,从最初的简单模式输入的直接的 SVM 方法研究,进入到多种方法取长补短的联合应用研究,对 SVM 方法也有了很多改进。

1. 人脸检测、验证和识别

Osuna 最早将 SVM 应用于人脸检测,并取得了较好的效果。其方法是直接训练非线性 SVM 分类器完成人脸与非人脸的分类。由于 SVM 的训练需要大量的存储空间,并且非线性 SVM 分类器需要较多的支持向量,速度很慢。为此,马勇等^①提出了一种层次型结构的 SVM 分类器,它由一个线性 SVM 组合和一个非线性 SVM 组成。检测时,由前者快速排除掉图像中绝大部分背景窗口,而后者只需对少量的候选区域做出确认;训练时,在线性 SVM 组合的限定下,与“自举(boot strapping)”方法相结合可收集到训练非线性 SVM 的更有效的非人脸样本,简化 SVM 训练的难度,大量实验结果表明这种方法不仅具有较高的检测率和较低的误检率,而且具有较快的速度。

人脸检测研究中更复杂的情况是姿态的变化。叶航军等^②提出了利用支持向量机方法进行人脸姿态的判定,将人脸姿态划分成 6 个类别,从一个多姿态人脸库中手工标定训练样本集和测试样本集,训练基于支持向量机姿态分类器,分类错误率降低到 1.67%。明显优于在传统方法中效果最好的人工神经网络方法。

在人脸识别中,面部特征的提取和识别可看作是对 3D 物体的 2D 投影图像进行匹配的问题。由于许多不确定性因素的影响,特征的选取与识别就成为一个难点。凌旭峰等^③及张燕昆等^④分别提出基于 PCA 与 SVM 相结合的人脸

①马勇,丁晓青. 基于层次型支持向量机的人脸检测[J]. 清华大学学报(自然科学版),2003,43(1):35-38.

②叶航军,白雪生,徐光祐. 基于支持向量机的人脸姿态判定[J]. 清华大学学报(自然科学版),2003,43(1):67-70.

③凌旭峰,杨杰,叶晨洲. 基于支撑向量机的人脸识别技术[J]. 红外与激光工程,2001,30(5):318-322.

④张燕昆,杨平,刘重庆. 基于主元分析与支持向量机的人脸识别方法[J]. 上海交通大学学报,2002,36(6):884-886.

识别算法,充分利用了 PCA 在特征提取方面的有效性以及 SVM 在处理小样本问题和泛化能力强等方面的优势,通过 SVM 与最近邻距离分类器相结合,使得所提出的算法具有比传统最近邻分类器和 BP 网络分类器更高的识别率。王宏漫等^①在 PCA 基础上进一步做 ICA,提取更加有利于分类的面部特征的主要独立成分;然后采用分阶段淘汰的支持向量机分类机制进行识别。对两组人脸图像库的测试结果表明,基于 SVM 的方法在识别率和识别时间等方面都取得了较好的效果。

2. 说话人/语音识别

说话人识别属于连续输入信号的分类问题,SVM 是一个很好的分类器,但不适合处理连续输入样本。为此,忻栋等^②引入隐式马尔可夫模型 HMM,建立了 SVM 和 HMM 的混合模型。HMM 适合处理连续信号,而 SVM 适合于分类问题;HMM 的结果反映了同类样本的相似度,而 SVM 的输出结果则体现了异类样本间的差异。为了方便与 HMM 组成混合模型,首先将 SVM 的输出形式改为概率输出。实验中使用 YOHO 数据库,特征提取采用 12 阶的线性预测系数分析及其微分,组成 24 维的特征向量。实验表明 HMM 和 SVM 的结合达到了很好的效果。

3. 文字/手写体识别

贝尔实验室对美国邮政手写数字库进行的实验,人工识别平均错误率是 2.5%,专门针对该特定问题设计的 5 层神经网络错误率为 5.1%(其中利用了大量先验知识),而用 3 种 SVM 方法(采用 3 种核函数)得到的错误率分别为 4.0%、4.1%和 4.2%,且是直接采用 16×16 的字符点阵作为输入,表明了 SVM 的优越性能。

手写体数字 0~9 的特征可以分为结构特征、统计特征等。柳回春等^③在

①王宏漫,欧宗瑛.采用 PCA/ICA 特征和 SVM 分类的人脸识别[J].计算机辅助设计与图形学学报,2003,15(4):416-420.

②忻栋,杨莹春.吴朝晖基于 SVM-HMM 混合模型的说话人确认[J].计算机辅助设计与图形学学报,2002,14(11):1080-1082.

③柳回春,马树元,吴平东.UK 心理测试自动分析系统的手写体数字识别,北京理工大学学报,2002,22(5):599-603.

UK 心理测试自动分析系统中组合 SVM 和其他方法成功地进行了手写数字的识别实验。另外,在手写汉字识别方面,高学等^①提出了一种基于 SVM 的手写汉字的识别方法,表明了 SVM 对手写汉字识别的有效性。

4. 图像处理

(1)图像过滤。一般的互联网色情网图像过滤软件主要采用网址库的形式来封锁色情网址或采用人工智能方法对接收到的中、英文信息进行分析甄别。段立娟等^②提出一种多层次特定类型图像过滤法,即以综合肤色模型检验,支持向量机分类和最近邻方法校验的多层次图像处理框架,达到 85% 以上的准确率。

(2)视频字幕提取。视频字幕蕴含了丰富语义,可用于对相应视频流进行高级语义标注。庄越挺等^③提出并实践了基于 SVM 的视频字幕自动定位和提取的方法。该方法首先将原始图像帧分割为 $N \times N$ 的子块,提取每个子块的灰度特征;然后使用预先训练好的 SVM 分类机进行字幕子块和非字幕子块的分类;最后结合金字塔模型和后期处理过程,实现视频图像字幕区域的自动定位提取。实验表明该方法取得了良好的效果。

(3)图像分类和检索。由于计算机自动抽取的图像特征和人所理解的语义间存在巨大的差距,图像检索结果难以令人满意。近年来出现了相关反馈方法,张磊等^④以 SVM 为分类器,在每次反馈中对用户标记的正例和反例样本进行学习,并根据学习所得的模型进行检索,使用由 9918 幅图像组成的图像库进行实验,结果表明,在有限训练样本情况下具有良好的泛化能力。

①高学,金连文,尹俊勋等一种基于支持向量机的手写汉字识别方法,电子学报,2002,30(5):651-654.

②段立娟,崔国勤,高文等多层次特定类型图像过滤方法,计算机辅助设计与图形学学报,2002,14(5):1-6.

③庄越挺,刘骏伟,吴飞等基于支持向量机的视频字幕自动定位与提取,计算机辅助设计与图形学学报,2002,14(8):1-4.

④张磊,林福宗,张钹基于支持向量机的相关反馈图像检索算法清华大学学报(自然科学版),200242(1):80-83.

目前 3D 虚拟物体图像应用越来越广泛,肖俊等^①提出了一种基于 SVM 对相似 3D 物体识别与检索的算法。该算法首先使用细节层次模型对 3D 物体进行三角面片数量的约减,然后提取 3D 物体的特征,由于所提取的特征维数很大,因此先用独立成分分析进行特征约减,然后使用 SVM 进行识别与检索。将该算法用于 3D 丘陵与山地的地形识别中,取得了良好效果。

5. 其他应用研究

(1) 由于 SVM 的优越性,其应用研究目前开展已经相当广泛。陈光英等^②设计并实现了一种基于 SVM 分类机的网络入侵检测系统。它收集并计算除服务器端口之外 TCP/IP 的流量特征,使用 SVM 算法进行分类,从而识别出该连接的服务类型,通过与该连接服务器端口所表明服务类型的比较,检测出异常的 TCP 连接。实验结果表明,系统能够有效地检测出异常 TCP 连接。

(2) 口令认证简便易实现,但容易被盗用。刘学军等^③提出利用 SVM 进行键入特性的验真,并通过实验将其与 BP、RBF、PNN 和 LVQ 4 种神经网络模型进行对比。证实了采用 SVM 进行键入特性验真的有效性。

(3) 李晓黎等^④提出了一种将 SVM 与无监督聚类相结合的新分类算法,并应用于网页分类问题。该算法首先利用无监督聚类分别对训练集中正例和反例聚类,然后挑选一些例子训练 SVM 并获得 SVM 分类器。任何网页可以通过比较其与聚类中心的距离决定采用无监督聚类方法或 SVM 分类器进行分类。该算法充分利用了 SVM 准确率高与无监督聚类速度快的优点。实验表明它不仅具有较高的训练效率,而且有很高的精确度。

(4) 刘江华等^⑤提出并实现一个用于人机交互的静态手势识别系统。基于

①肖俊,吴飞,庄越挺等基于支持向量机与细节层次的三维地形识别与检索[J]. 计算机辅助设计与图形学学报,2003,15(4):61-63.

②陈光英,张千里,李星. 基于 SVM 分类机的入侵检测系统[J]. 通信学报,2002,23(5):51-56.

③刘学军,陈松灿,彭宏京. 基于支持向量机的计算机键盘用户身份验真[J]. 计算机研究与发展,2002,39(9):1082-1086.

④李晓黎,刘继敏,史忠植. 基于支持向量机与无监督聚类相结合的中文网页分类器[J]. 计算机学报,2001,24(1):62-68.

皮肤颜色模型进行手势分割,并用傅立叶描述子描述轮廓,采用最小二乘支持向量机(LS-SVM)作为分类器。提出了LS-SVM的增量训练方式,避免了费时的矩阵求逆操作。为实现多类手势识别,利用DAG(Directed Acyclic Graph)将多个两类LS-SVM结合起来。对26个字母手势进行识别。与多层感知器、径向基函数网络等方法比较,LS-SVM的识别率最高,达到93.62%。

另外的研究还有应用SVM进行文本分类、应用SVM构造自底向上二叉树结构进行空间数据聚类分析等。近年来,SVM在工程实践、化学化工等方面也取得了很多有益的应用研究成果,其应用领域日趋广泛。

①刘江华,陈佳品,程君.实用于人机交互的静态手势识别系统[J].红外与激光工程,2002,31(6):499-503.

第 5 章 机器学习方法

5.1 机器学习的历史

自从 20 世纪 50 年代开始研究机器学习以来,在不同时期的研究途径和目标也不同,可以划分为四个阶段。

第一阶段是 20 世纪 50 年代中叶到 60 年代中叶,属于热烈时期。在这个时期,所研究的是“没有知识”的学习,即“无知”学习,它的主要研究目标是各类自组织系统和自适应系统。例如,如果给系统一组刺激、一个反馈源以及修改它们自身组织的足够自由度,那么它们将改变自身成为最优的组织,即它们能够修改自身以适应它们的环境。这类系统采用的主要研究方法是不断修改系统的控制参数和改进系统的执行能力,不涉及与具体任务有关的知识。本阶段的代表性工作是 Samuel 的下棋程序。但这种学习的结果远不能满足人们对机器学习系统的期望。

第二阶段是在 20 世纪 60 年代中叶到 70 年代中叶,被称为机器学习的冷静时期。本阶段的研究目标是模拟人类的概念学习过程,并采用逻辑结构或图结构作为机器内部描述。机器采用符号来表示概念,又称符号概念获取,并提出关于所学概念的各种假设。在这一阶段,研究者意识到学习是复杂而困难的过程,因此人们不能期望学习系统可以从没用任何知识的环境中开始,学习到高深而有价值的概念。这种观点使得研究人员一方面深入探讨简单的学习问题,另一方面则把大量的领域专家知识加入到学习系统中。本阶段的代表性工作有 Winston 的结构学习系统和 Hayes Roth 等的基于逻辑的归纳学习系统。

第三阶段从 20 世纪 70 年代中叶到 80 年代中叶,称为复兴时期。在此期间,人们从学习单个概念扩展到学习多个概念,探索不同的学习策略和学习方法,且在本阶段已开始把学习系统与各种应用结合起来,并取得很大的成功。同时,专家系统在知识获取方面的需求,也极大地刺激了机器学习的研究和发展。在出现第一个专家学习系统之后,示例归纳学习系统成为研究的主流,自动知识获取成为机器学习应用的研究目标。1980 年,在美国的卡内基-梅隆

(CMU)召开了第一届机器学习国际研讨会,标志着机器学习研究已在全世界兴起。此后,机器学习开始得到了大量的应用。Strategic Analysis and Information System 国际杂志连续三期刊登有关机器学习的文章。1984年,由Simon等20多位人工智能专家共同撰文编写的Machine Learning文集第二卷出版,国际性杂志Machine Learning创刊,更加显示出机器学习突飞猛进的发展趋势。这一阶段代表性的工作有Mostow的指导式学习,Lenat的数学概念发现程序,Langley的BACON程序及其改进程序。

第四阶段从20世纪80年代中叶到现在,是机器学习的最新阶段。这个时期的机器学习具有如下特点:机器学习已成为新的边缘学科,它综合应用了心理学、生物学和神经生理学以及数学、自动化和计算机科学形成机器学习理论基础;融合了各种学习方法,且形式多样的集成学习系统研究正在兴起;机器学习与人工智能各种基础问题的统一性观点正在形成;各种学习方法的应用范围不断扩大,一部分应用研究成果已转化为商品;与机器学习有关的学术活动空前活跃。

5.2 监督学习

5.2.1 未标记样本

我们在丰收季节来到瓜田,满地都是西瓜,瓜农抱来三四个瓜说这都是好瓜,然后再指着地里的五六个瓜说这些还不好,还需再生长若干天。基于这些信息,我们能否构建一个模型,用于判别地里的哪些瓜是已该采摘的好瓜?显然,可将瓜农告诉我们的好瓜、不好的瓜分别作为正例和反例来训练一个分类器。然而,只用这不到十个瓜做训练样本,有点太少了?能不能把地里的那些瓜也用上呢?

形式化地看,我们有训练样本集 $D_l = \{(x_1, y_1), (x_2, y_2), \dots, (x_l, y_l)\}$, 这 l 个样本的类别标记(即是否好瓜)已知,称为“有标记”(labeled)样本;此外,还有 $D_u = \{x_l + 1, x_l + 2, \dots, x_l + u\}$, $l < u$, 这 u 个样本的类别标记未知(即不知是否好瓜),称为“未标记”(unlabeled)样本。一方面,若直接使用传统监督学习技术,则仅有功能用于构建模型, D_u 所包含的信息被浪费了;另一方面,若以较

小,则由于训练样本不足,学得模型的泛化能力往往不佳。那么,能否在构建模型的过程中将 D_u 利用起来呢?

一个简单的做法,是将中的示例全部标记后用于学习。这就相当于请瓜农把地里的瓜全都检查一遍,告诉我们哪些是好瓜,哪些不是好瓜,然后再用于模型训练。显然,这样做需耗费瓜农大量时间和精力。有没有“便宜”一点的办法呢?

我们可以用先训练一个模型,拿这个模型去地里挑一个瓜,询问瓜农好不好,然后把这个新获得的有标记样本加入 D 中重新训练一个模型,再去挑瓜,……这样,若每次都挑出对改善模型性能帮助大的瓜,则只需询问瓜农比较少的瓜就能构建出比较强的模型,从而大幅降低标记成本。这样的学习方式称为“主动学习”(active learning),其目标是使用尽量少的“查询”(query)来获得尽量好的性能。

显然,主动学习引入了额外的专家知识,通过与外界的交互来将部分未标记样本转变为有标记样本。若不与专家交互,没有获得额外信息,还能利用未标记样本来提高泛化性能吗?

答案是“Yes!”,有点匪夷所思?

事实上,未标记样本虽未直接包含标记信息,但若它们与有标记样本是从同样的数据源独立同分布采样而来,则它们所包含的关于数据分布的信息对建立模型将大有裨益。图 5-1 给出了一个直观的例示。若仅基于图中的一个正例和一个反例,则由于待判别样本恰位于两者正中间,大体上只能随机猜测;若能观察到图中的未标记样本,则将很有把握地判别为正例。

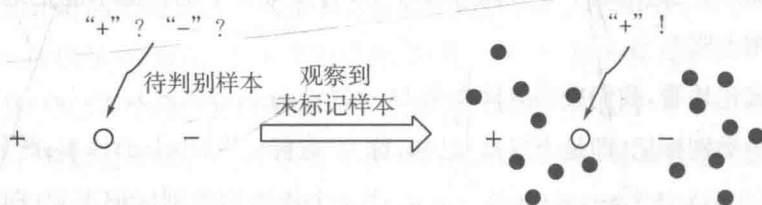


图 5-1 未标记样本效用的示例(右边的灰色点表示未标记样本)

让学习器不依赖外界交互、自动地利用未标记样本来提升学习性能,就是

半监督学习(Semi-Supervised Learning)^①。半监督学习的现实需求非常强烈,因为在现实应用中往往能容易地收集到大量未标记样本,而获取“标记”却需耗费人力、物力。例如,在进行计算机辅助医学影像分析时,可以从医院获得大量医学影像,但若希望医学专家把影像中的病灶全都标识出来则是不现实的。“有标记数据少,未标记数据多”这个现象在互联网应用中更明显,例如在进行网页推荐时需请用户标记出感兴趣的网页,但很少有用户愿花很多时间来提供标记,因此,有标记网页样本少,但互联网上存在无数网页可作为未标记样本来使用。半监督学习恰是提供了一条利用“廉价”的未标记样本的途径。

要利用未标记样本,必然要做一些将未标记样本所揭示的数据分布信息与类别标记相联系的假设。最常见的是“聚类假设”(cluster assumption),即假设数据存在簇结构,同一个簇的样本属于同一个类别。由于待预测样本与正例样本通过未标记样本的“撮合”聚在一起,与相对分离的反例样本相比,待判别样本更可能属于正类。半监督学习中另一种常见的假设是“流形假设”(manifold assumption),即假设数据分布在一个流形结构上,邻近的样本拥有相似的输出值“邻近”程度常用“相似”程度来刻画,因此,流形假设可看作聚类假设的推广,但流形假设对输出值没有限制,因此比聚类假设的适用范围更广,可用于更多类型的学习任务。事实上,无论聚类假设还是流形假设,其本质都是“相似的样本拥有相似的输出”这个基本假设。

半监督学习可进一步划分为纯半监督学习(pure)和直推学习(transductive learning),前者假定训练数据中的未标记样本并非待预测的数据,而后者则假定学习过程中所考虑的未标记样本恰是待预测数据,学习的目的就是在这些未标记样本上获得最优泛化性能。换言之,纯半监督学习是基于“开放世界”假设,希望学得模型能适用于训练过程中未观察到的数据;而直推学习是基于“封闭世界”假设,仅试图对学习过程中观察到的未标记数据进行预测。图5-2直观地显示出主动学习、纯半监督学习、直推学习的区别。需注意的是纯半监督学习和直推学习常合称为半监督学习,本书也采取这一态度,在需专门区分时会特别说明。

①刘建伟,刘媛,罗雄麟. 半监督学习方法[J]. 计算机学报,2015(8):1592-1617.

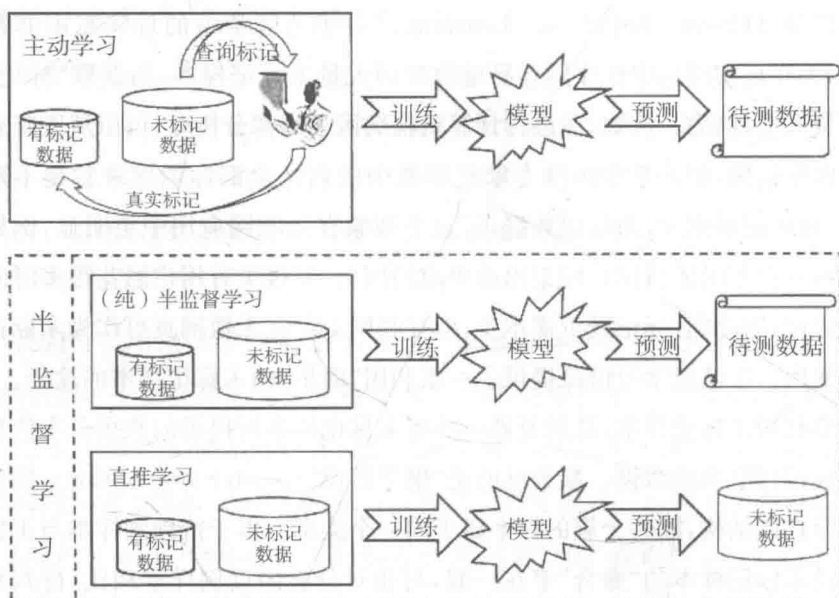


图 5-2 主动学习、(纯)半监督学习、直推学习

5.2.2 生成式方法

生成式方法 (generative methods) 是直接基于生成式模型的方法。此类方法假设所有数据 (无论是否有标记) 都是由同一个潜在的模型“生成”的, 这个假设使得我们能通过潜在模型的参数将未标记数据与学习目标联系起来, 而未标记数据的标记则可看作模型的缺失参数, 通常可基于 EM 算法进行极大似然估计求解。此类方法的区别主要在于生成式模型的假设, 不同的模型假设将产生不同的方法。

给定样本 x , 其真实类别标记为 $y \in Y$, 其中 $Y = \{1, 2, \dots, N\}$ 为所有可能的类别。假设样本由高斯混合模型生成, 且每个类别对应一个高斯混合成分。换言之, 数据样本是基于如下概率密度生成:

$$p(x) = \sum_{i=1}^N \alpha_i \cdot p(x | \mu_i, \Sigma_i) \quad (5-1)$$

其中, 混合系数 $\alpha_i \geq 0$, $\sum_{i=1}^N \alpha_i = 1$; $p(x_i | \mu_i, \Sigma_i)$ 是样本 x 属于第 i 个高斯混合成分的概率; μ_i 和 Σ_i 为该高斯混合成分的参数。

令 $f(x) \in Y$ 表示模型 f 对 x 的预测标记, $\Theta \in \{1, 2, \dots, N\}$ 表示样本 x 隶属的高斯混合成分。由最大化后验概率可知

$$\begin{aligned}
 f(x) &= \underset{j \in y}{\operatorname{argmax}} p(j|x) = \underset{j \in y}{\operatorname{argmax}} \sum_{i=1}^N p(y=j, \Theta=i|x) \\
 &= \underset{j \in y}{\operatorname{argmax}} \sum_{i=1}^N p(y=j|\Theta=i, x) \cdot P(\Theta=i|x)
 \end{aligned} \quad (5-2)$$

其中

$$P(\Theta=i|x) = \frac{\alpha_i \cdot p(x|\mu_i, \Sigma_i)}{\sum_{i=1}^N \alpha_i \cdot p(x|\mu_i, \Sigma_i)} \quad (5-3)$$

为样本 x 由第 i 个高斯混合成分生成的后验概率, $p(y=j | \Theta=i, x)$ 为 x 由第 i 个高斯混合成分生成且其类别为 j 的概率。由于假设每个类别对应一个高斯混合成分, 因此 $p(y=j | \Theta=i, x)$ 仅与样本 x 所属的高斯混合成分 Θ 有关, 可用 $p(y=j | \Theta=i, x)$ 代替。不失一般性, 假定第 i 个类别对应于第 i 个高斯混合成分, 即 $p(y=j|\Theta=i)=1$ 当且仅当 $i=j$, 否则 $p(y=j|\Theta=i)=0$ 。

不难发现, 式(5-2)中估计, $p(y=j | \Theta=i, x)$ 需知道样本的标记, 因此仅能使用有标记数据; 而 $p(\Theta=i | x)$ 不涉及样本标记, 因此有标记和未标记数据均可利用, 通过引入大量的未标记数据, 对这一项的估计可望由于数据量的增长而更为准确, 于是式(5-2)整体的估计可能会更准确。由此可清楚地看出未标记数据何以能辅助提高分类模型的性能。

给定有标记样本集以 $D_l = \{(x_1, y_1), (x_2, y_2), \dots, (x_l, y_l)\}$ 和未标记样本集 $\{x_{l+1}, x_{l+2}, \dots, x_{l+u}\}$, $l < u$, $l+u=m$ 。假设所有样本独立同分布, 且都是由同一个高斯混合模型生成的, 用极大似然法来估计高斯混合模型的参数 $\{\alpha_i, u_i, \sum_i | 1 < i < N\}$, $D_l U D_u$ 的对数似然是

$$\begin{aligned}
 LL(D_l U D_u) &= \sum_{(x_j, y_j) \in D_l} \ln \left(\sum_{i=1}^N \alpha_i \cdot p(x_j|\mu_i, \Sigma_i) \cdot p(y_j|\Theta=i, x_j) \right) \\
 &\quad + \sum_{x_j \in D_u} \ln \left(\sum_{i=1}^N \alpha_i \cdot p(x_j|\mu_i, \Sigma_i) \right)
 \end{aligned} \quad (5-4)$$

式(5-4)由两项组成: 基于有标记数据的有监督项和基于未标记数据的无监督项。显然, 高斯混合模型参数估计可用 EM 算法求解, 迭代更新式如下。

E 步: 根据当前模型参数计算未标记样本 % 属于各高斯混合成分的概率

$$\gamma_{ij} = \frac{\alpha_i \cdot p(x_j|\mu_i, \Sigma_i)}{\sum_{i=1}^N \alpha_i \cdot P(x_j|\mu_i, \Sigma_i)} \quad (5-5)$$

M 步: 基于 γ_{ji} 更新模型参数, 其中 L_i 表示第 i 类的有标记样本数目

$$\mu_i = \frac{1}{\sum_{x_j \in D_u} \gamma_{ij} + l_i} \left(\sum_{x_j \in D_u} \gamma_{ij} x_j + \sum_{x_j, y_j \in D_l \wedge y_j = i} x_j \right) \quad (5-6)$$

$$\Sigma_i = \frac{1}{\sum_{x_j \in D_u} \gamma_{ij} + l_i} \left(\sum_{x_j \in D_u} \gamma_{ij} (x_j - u_i)(x_j - u_i)^T + \sum_{x_j, y_j \in D_l \wedge y_j = i} (x_j - u_i)(x_j - u_i)^T \right) \quad (5-7)$$

$$\alpha_i = \frac{1}{m} \left(\sum_{x_j \in D_u} \gamma_{ij} + l_i \right) \quad (5-8)$$

以上过程不断迭代直至收敛,即可获得模型参数。然后由式(5-3)和(5-2)就能对样本进行分类。

将上述过程中的高斯混合模型换成混合专家模型(MillerandUyar, 1997)、朴素贝叶斯模型(Nigametal, 2000)等即可推导出其他的生成式半监督学习。此类方法简单,易于实现,在有标记数据极少的情形下往往比其他方法性能更好。然而,此类方法有一个关键:模型假设必须准确,即假设的生成式模型必须与真实数据分布吻合;否则利用未标记数据反而会降低泛化性能(Cozmanand-Cohen, 2002),遗憾的是,在现实任务中往往很难事先做出准确的模型假设,除非拥有充分可靠的领域知识。

5.2.3 阅读材料

半监督学习的研究一般认为始于 Shahshahani and Landgrebe(1994)^①,该领域在 20 世纪末、21 世纪初随着现实应用中利用未标记数据的巨大需求涌现而蓬勃发展。国际机器学习大会(ICML)从 2008 年开始评选“十年最佳论文”,在短短 6 年中,半监督学习四大模型(paradigm)中基于分歧的方法、半监督 SVM、图半监督学习的代表性工作先后于 2008 年(Plumand Mitchell, 1998)、2009 年(Joachims, 1999)、2013 年(Zhu et al, 2003)获奖。生成式半监督学习方法出现最早(Shahshahani and Landgrebe, 1994)。由于需有充分可靠的领域知识才能确保模型假设不至于太坏,因此该模型后来主要是在具体的应用领域加

①BELKIN MP, NIYOGI, SINDHWANI V. Manifold regularization: A geometric frame work for learning from labeled and unlabeled examples[J]. Journal of Machine Learning Research, 2006(7). 7:2399-2434.

以研究。

半监督 SVM 的目标函数非凸,有不少工作致力于减轻非凸性造成的不利影响,例如使用连续统(continuation)方法,从优化一个简单的图目标函数开始,逐步变形为非凸的 S3VM 目标函数(Chapelle et al, 2006);使用确定性退火(deterministic annealing)过程,将非凸问题转化为一系列凸优化问题,然后由易到难的求解(Sindhwani et al, 2006);利用 CCCP 方法优化非凸函数(Collobert et al, 2006)等。

基于分歧的方法起源于协同训练,最初设计是仅选取一个学习器用于预测(Blum and Mitchell, 1998)。三体训练(tri-training)使用三个学习器,通过“少数服从多数”来产生伪标记样本,并将学习器进行集成(Zhou and Li, 2005)。后续研究进一步显示出将学习器集成起来更有助于性能提升,并出现了使用更多学习器的方法。更为重要的是,这将集成学习与半监督学习这两个长期独立发展的领域联系起来(Zhou, 2009)。此外,这些方法能容易地用于多视图数据,并可自然地与主动学习进行结合(周志华, 2013)。

(Belkin et al, 2006)在半监督学习中提出了流形正则化(manifold regularization)框架,直接基于局部光滑性假设对定义在有标记样本上的损失函数进行正则化,使学得预测函数具有局部光滑性。

半监督学习在利用未标记样本后并非必然提升泛化性能,在有些情形下甚至会导致性能下降。对生成式方法,其成因被认为是模型假设不准确(Cozman and Cohen, 2002),因此需依赖充分可靠的领域知识来设计模型。对半监督 SVM,其成因被认为是训练数据中存在多个“低密度划分”,而学习算法有可能做出不利的选择;S4VM(Li and Zhou, 2015)通过优化最坏情形性能来综合利用多个低密度划分,提升了此类技术的安全性。更一般的“安全”(safe)半监督学习仍是一个未决问题。

5.3 集成学习

5.3.1 个体与集成

集成学习(ensemble learning)通过构建并结合多个学习器来完成学习任

务,有时也被称为多分类器系统(multi-classifier system)、基于委员会的学习(committee-based learning)等。

图 5-3 显示出集成学习的一般结构:先产生一组“个体学习器”(individual learner),再用某种策略将它们结合起来。个体学习器通常由一个现有的学习算法从训练数据产生,例如决策树算法、BP 神经网络算法等,此时集成中只包含同种类型的个体学习器,例如“决策树集成”中全是决策树,“神经网络集成”中全是神经网络,这样的集成是“同质”的(homogeneous)。同质集成中的个体学习器亦称“基学习器”(baselearner),相应的学习算法称为“基学习算法”(baselearning algorithm)。集成也可包含不同类型的个体学习器,例如同时包含决策树和神经网络,这样的集成是“异质”的(heterogeneous)。异质集成中的个体学习器由不同的学习算法生成,这时就不再有基学习算法;相应的个体学习器一般不称为基学习器,常称为“组件学习器”(component learner)或直接称为个体学习器。

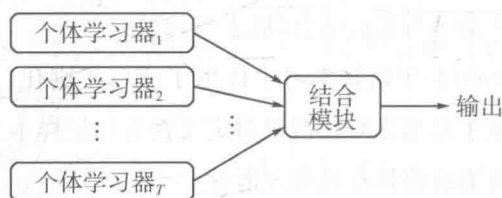


图 5-3 集成学习示意图

集成学习通过将多个学习器进行结合,常可获得比单一学习器显著优越的泛化性能,这对“弱学习器”(weak learner)尤为明显,因此集成学习的很多理论研究都是针对弱学习器进行的,而基学习器有时也被直接称为弱学习器。但需注意的是,虽然从理论上来说使用弱学习器集成足以获得好的性能,但在实践中出于种种考虑,例如希望使用较少的个体学习器,或是重用关于常见学习器的一些经验等,人们往往会使用比较强的学习器。

在一般经验中,如果把好坏不等的东西掺到一起,那么通常结果会是比最坏的要好一些,比最好的要坏一些。集成学习把多个学习器结合起来,如何能获得比最好的单一学习器更好的性能呢?

考虑一个简单的例子:在二分类任务中,假定三个分类器在三个测试样本

上的表现如图 5-4 所示,其中 V 表示分类正确, x 表示分类错误,集成学习的结果通过投票法(voting)产生,即“少数服从多数”。在图 5-4(a)中,每个分类器都只有 66.6%的精度,但集成学习却达到了 100%;在图 5-4(b)中,三个分类器没有差别,集成之后性能没有提高;在图 5-4(c)中,每个分类器的精度都只有 33.3%,集成学习的结果变得更糟。这个简单的例子显示出:要获得好的集成,个体学习器应“好而不同”,即个体学习器要有一定的“准确性”,即学习器至少不差也不能太坏,并且要有“多样性”(diversity),即学习器间具有差异。

测 测 测 试 试 试 例 1 例 2 例 3				测 测 测 试 试 试 例 1 例 2 例 3				测 测 测 试 试 试 例 1 例 2 例 3			
h_1	✓	✓	×	h_1	✓	✓	×	h_1	✓	×	×
h_2	×	✓	✓	h_2	✓	✓	×	h_2	×	✓	×
h_3	✓	×	✓	h_3	✓	✓	×	h_3	×	×	✓
集 成	✓	✓	✓	集 成	✓	✓	×	集 成	×	×	×
(a)集成提升性能				(b)集成不起作用				(c)集成起负作用			

图 5-4 集成个体应“好而不同”(h_i 表示第 i 个分类器)

我们来做个简单的分析,考虑二分类问题 $y \in \{-1, +1\}$ 和真实函数 f ,假定基分类器的错误率为 ϵ ,即对每个基分类器 h_i 有

$$P(h_i(x) \neq f(x)) = \epsilon \quad (5-9)$$

假设集成通过简单投票法结合 T 个基分类器,若有超过半数的基分类器正确,则集成分类就正确:

$$H(x) = \text{sign}\left(\sum_{i=1}^T h_i(x)\right) \quad (5-10)$$

假设基分类器的错误率相互独立,则由 Hoeffding 不等式可知,集成的错误率为

$$P(H(x) \neq f(x)) = \left(\sum_{k=0}^{T/2} \binom{T}{k} (1-\epsilon)^k \epsilon^{T-k}\right) \leq \exp(-1/2T(1-2\epsilon)^2) \quad (5-11)$$

上式显示出,随着集成中个体分类器数目 T 的增大,集成的错误率将指数级下降,最终趋向于零。

然而我们必须注意到,上面的分析有一个关键假设:基学习器的误差相互独立。在现实任务中,个体学习器是为解决同一个问题训练出来的,它们显然不可能相互独立!事实上,个体学习器的“准确性”和“多样性”本身就存在冲突。一般的,准确性很高之后,要增加多样性就需牺牲准确性。事实上,如何产生并结合“好而不同”的个体学习器,恰是集成学习研究的核心。

根据个体学习器的生成方式,目前的集成学习方法大致可分为两大类,即个体学习器间存在强依赖关系、必须串行生成的序列化方法,以及个体学习器间不存在强依赖关系、可同时生成的并行化方法;前者的代表是 Boosting,后者的代表是 Bagging 和“随机森林”(Random Forest)。

5.3.2 Bagging 与随机森林

由 5.3.1 节可知,欲得到泛化性能强的集成,集成中的个体学习器应尽可能相互独立;虽然“独立”在现实任务中无法做到,但可以设法使基学习器尽可能具有较大的差异。给定一个训练数据集,一种可能的做法是对训练样本进行采样,产生出若干个不同的子集,再从每个数据子集中训练出一个基学习器。这样,由于训练数据不同,我们获得的基学习器可望具有比较大的差异。然而,为获得好的集成,我们同时还希望个体学习器不能太差。如果采样出的每个子集都完全不同,则每个基学习器只用到了一小部分训练数据,甚至不足以进行有效学习,这显然无法确保产生出比较好的基学习器。为解决这个问题,我们可考虑使用相互有交叠的采样子集。

1. Bagging

Bagging(Breiman,1996)是并行式集成学习方法最著名的代表。从名字即可看出,它直接基于自助采样法(Bootstrap Sampling)。给定包含 m 个样本的数据集,我们先随机取出一个样本放入采样集中,再把该样本放回初始数据集,使得下次采样时该样本仍有可能被选中,这样,经过 m 次随机采样操作,我们得到含 m 个样本的采样集,初始训练集中有的样本在采样集里多次出现。

照这样,我们可采样出 T 个含 m 个训练样本的采样集,然后基于每个采样

集训练出一个基学习器,再将这些基学习器进行结合。这就是 Bagging 的基本流程。在对预测输出进行结合时, Bagging 通常对分类任务使用简单投票法,对回归任务使用简单平均法。若分类预测时出现两个类收到同样票数的情形,则最简单的做法是随机选择一个,也可进一步考察学习器投票的置信度来确定最终胜者。Bagging 的算法描述如图 5-5 所示。

输入: 训练集 $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$;

基学习算法 ϵ ;

训练轮数 T ;

过程:

1: for $t = 1, 2, \dots, T$ do

2: $h_t = \epsilon(D, D_{bs})$

3: endfor

输出: $H(x) = \operatorname{argmax}_{y \in Y} \sum_{t=1}^T \Pi(h_t(x) = y)$

图 5-5 Bagging 算法

假定基学习器的计算复杂度为 $O(m)$, 则 Bagging 的复杂度大致为 $T(O(m) + O(s))$, 考虑到采样与投票/平均过程的复杂度 $O(s)$ 很小, 而 T 通常是一个不太大的常数, 因此, 训练一个 Bagging 集成与直接使用基学习算法训练一个学习器的复杂度同阶, 这说明 Bagging 是一个很高效的集成学习算法。另外, 与标准 AdaBoost 只适用于二分类任务不同, Bagging 能不经修改地用于多分类、回归等任务。

值得一提的是, 自助采样过程还给 Bagging 带来了另一个优点: 由于每个基学习器只使用了初始训练集中约 63.2% 的样本, 剩下约 36.8% 的样本可用作验证集来对泛化性能进行“包外估计”(Out of Bag estimate) (Breiman, 1996; Wolpert and Macready, 1999)。为此需记录每个基学习器所使用的训练样本。不妨令 D_t 表示 h_t 实际使用的训练样本集, 令 $Hoob(x)$ 表示对样本 x 的包外预测, 即仅考虑那些未使用 x 训练的基学习器在 x 上的预测, 有

$$Hoob(x) = \operatorname{argmax}_{y \in Y} \sum_{t=1}^T \Pi(h_t(x) = y) \cdot \Pi(x \cdot D_t), \quad (5-12)$$

则 Bagging 泛化误差的外包估计为

$$\epsilon_{oob} = \frac{1}{|D|} \sum_{(x,y) \in D} \Pi(H_{oob}(x) \neq y) \quad (5-13)$$

事实上,包外样本还有许多其他用途,例如当基学习器是决策树时,可使用包外样本来辅助剪枝,或用于估计决策树中各节点的后验概率以辅助对零训练样本节点的处理;当基学习器是神经网络时,可使用包外样本来辅助早期停止以减小过拟合风险。

从偏差-方差分解的角度看, Bagging 主要关注降低方差,因此它在不剪决策树、神经网络等易受样本扰动的学习器上效用更为明显。

2、机森林

随机森林(Random Forest, RF)(Breiman, 2001)是 Bagging 的一个扩展变体。RF 在以决策树为基学习器构建 Bagging 集成的基础上,进一步在决策树的训练过程中引入了随机属性选择。具体来说,传统决策树在选择划分属性时是在当前节点的属性集合(假定有 d 个属性)中选择一个最优属性;而在 RF 中,对基决策树的每个节点,先从该节点的属性集合中随机选择一个包含 k 个属性的子集,然后再从这个子集中选择一个最优属性用于划分。这里的参数 k 控制了随机性的引入程度:若令 $k=1$,则基决策树的构建与传统决策树相同;若令 $k=d$,则是随机选择一个属性用于划分;一般情况下,推荐值 $k=\log_2 d$ (Breiman, 2001)。

随机森林简单、容易实现、计算开销小,令人惊奇的是,它在很多现实任务中展现出强大的性能,被誉为“代表集成学习技术水平的方法”可以看出,随机森林对 Bagging 只做了小改动,但是与 Bagging 中基学习器的“多样性”仅通过样本扰动(通过对初始训练集采样)而来不同,随机森林中基学习器的多样性不仅来自样本扰动,还来自属性扰动,这就使得最终集成的泛化性能可通过个体学习器之间差异度的增加而进一步提升。

随机森林的收敛性与 Bagging 相似。随机森林的起始性能往往相对较差,特别是在集成中只包含一个基学习器时。这很容易理解,因为通过引入属性扰动,随机森林中个体学习器的性能往往有所降低。然而,随着个体学习器数目的增加,随机森林通常会收敛到更低的泛化误差。值得一提的是,随机森林的

训练效率常优于 Bagging,因为在个体决策树的构建过程中, Bagging 使用的是“确定型”决策树,在选择划分属性时要对节点的所有属性进行考察,而随机森林使用的“随机型”决策树则只需考察一个属性子集。

5.4 规则学习

5.4.1 基本概念

机器学习中的“规则”(rule)通常是指语义明确、能描述数据分布所隐含的客观规律或领域概念、可写成“若……,则……”形式的逻辑规则 (Firnkrantz et al, 2012)。“规则学习”(rule learning)是从训练数据中学习出一组能用于对未见示例进行判别的规则。

形式化地看,一条规则形如:

$$\leftarrow f_1 \wedge f_1 \wedge \cdots \wedge f_i \quad (5-14)$$

其中,逻辑蕴含符号“ $\leftarrow$ ”右边部分称为“规则体”(body),表示该条规则的前提,左边部分称为“规则头”(head),表示该条规则的结果。规则体是由逻辑文字(literal) f_k 组成的合取式(conjunction),其中合取符号“ $\wedge$ ”用来表示“并且”。每个文字 f_k 都是对示例属性进行检验的布尔表达式,例如“(色泽=乌黑)”或“(根蒂=硬挺)”。 L 是规则体中逻辑文字的个数,称为规则的长度。规则头的“+? +”同样是逻辑文字,一般用来表示规则所判定的目标类别或概念,例如“好瓜”。这样的逻辑规则也被称为“if-then 规则”。

一方面,与神经网络、支持向量机这样的“黑箱模型”相比,规则学习具有更好的可解释性,能使用户更直观地对判别过程有所了解;另一方面,数理逻辑具极强的表达能力,绝大多数人类知识都能通过数理逻辑进行简洁的刻画和表达。例如“父亲的父亲是爷爷”这样的知识不易用函数式描述,而用一阶逻辑则可方便地写为“ $\text{爷爷}(x, r) \wedge \text{父亲}(X, Z) \wedge \text{父亲}(Z, Y)$ ”,因此,规则学习能更自然地在学习过程中引入领域知识。此外,逻辑规则的抽象描述能力在处理一些高度复杂的 AI 任务时具有显著的优势,例如在问答系统中有时可能遇到非常多、甚至无穷种可能的答案,此时若能基于逻辑规则进行抽象表述或者推理,则将带来极大的便利。

假定我们从西瓜数据集学得规则集合 R :

规则 1: 好瓜 $\leftarrow (\text{根蒂} = \text{蜷缩}) \wedge (\text{脐部} = \text{凹陷})$;

规则 2: $\neg$ 好瓜 $\leftarrow (\text{纹理} = \text{模糊})$ 。

规则 1 的长度为 2, 它通过判断两个逻辑文字的赋值(valuation)来对示例进行判别。符合该规则的样本(例如西瓜数据集 2.0 中的样本 1)称为被该规则“覆盖”(cover)。需注意的是, 被规则 1 覆盖的样本是好瓜, 但没被规则 1 覆盖的未必不是好瓜; 只有被规则 2 这样以好瓜为头的规则覆盖的才不是好瓜。

显然, 规则集合中的每条规则都可看作一个子模型, 规则集合是这些子模型的一个集成。当同一个示例被判别结果不同的多条规则覆盖时, 称发生了“冲突”(conflict), 解决冲突的办法称为“冲突消解”(conflict resolution)。常用的冲突消解策略有投票法、排序法、元规则法等。投票法是将判别相同的规则数最多的结果作为最终结果。排序法是在规则集合上定义一个顺序, 在发生冲突时使用排序最前的规则; 相应的规则学习过程称为“带序规则”(ordered rule)学习或“优先级规则”(priority rule)学习。元规则法是根据领域知识事先设定一些“元规则”(meta-rule), 即关于规则的规则, 例如“发生冲突时使用长度最小的规则”, 然后根据元规则的指导来使用规则集。

此外, 从训练集学得的规则集合也许不能覆盖所有可能的未见示例, 例如前述规则集合就无法对“根蒂 = 蜷缩”、“脐部 = 稍凹”且“纹理 = 清晰”的示例进行判别, 这种情况在属性数目很多时常出现。因此, 规则学习算法通常会设置一条“默认规则”(default rule), 由它来处理规则集合未覆盖的样本, 例如为 R 增加一条默认规则: “未被规则 1, 2 覆盖的都不是好瓜”。

从形式语言表达能力而言, 规则可分为两类: “命题规则”(propositional-rule)和“一阶规则”(first-order rule)。前者是由“原子命题”(propositional atom)和逻辑连接词“与”($\wedge$)、“或”($\vee$)、“非”($\neg$)和“蕴含”($\leftarrow$)构成的简单陈述句; 例如规则集兄就是一个命题规则集, “根蒂 = 蜷缩”“脐部 = 凹陷”都是原子命题, 后者的基本成分是能描述事物的属性或关系的“原子公式”(atomic formula), 例如表达父子关系的谓词(predicate)“父亲(X, Y)”就是原子公式, 再如表示加一操作“ $\sigma(X) = X + 1$ ”的函数“ $\sigma(X)$ ”也是原子公式。如果进一步用谓词“自然数(X)”表示 X 是自然数, “ $\forall X$ ”表示“对于任意 X 成立”, “ $\exists Y$ ”表示

“存在 Y 使之成立”,那么“所有自然数加 1 都是自然数”就可写作“ $\forall X \exists Y$ (自然数(Y) $\leftarrow$ 自然数(X) $\wedge$ ($Y = \sigma(X)$))”,或更简洁的“ $\forall X$ (自然数($\sigma(X)$) $\leftarrow$ 自然数(X))”。这样的规则就是一阶规则,其中 X 和 Y 称为逻辑变量,“ $\forall$ ”“ $\exists$ ”分别表示“任意”和“存在”,用于限定变量的取值范围,称为“量词”(quantifier)。显然,一阶规则能表达复杂的关系,因此也被称为“关系型规则”(relational rule)。以西瓜数据为例,若我们简单地把属性当作谓词来定义示例与属性值之间的关系,则命题规则集 R 可改写为一阶规则集 R' :

规则 1: 好瓜(X) $\leftarrow$ 根蒂(X , 蜷缩) $\wedge$ 脐部(X , 凹陷);

规则 2: $\neg$ 好瓜(X) $\leftarrow$ 纹理(X , 模糊)。

显然,从形式语言系统的角度来看,命题规则是一阶规则的特例,因此一阶规则的学习比命题规则要复杂得多。

5.4.2 序贯覆盖

规则学习的目标是产生一个能覆盖尽可能多的样例的规则集。最直接的做法是“序贯覆盖”(sequential covering),即逐条归纳:在训练集上每学到一条规则,就将该规则覆盖的训练样例去除,然后以剩下的训练样例组成训练集重复上述过程。由于每次只处理一部分数据,因此也被称为“分治”(separate-and-conquer)策略。

我们以命题规则学习为例来考察序贯覆盖法。命题规则的规则体是对样例属性值进行评估的布尔函数,如“色泽=青绿”、“含糖率 ≤ 0.2 ”等,规则头是样例类别。序贯覆盖法的关键是如何从训练集学出单条规则。显然,对规则学习目标“+?”,产生一条规则就是寻找最优的一组逻辑文字来构成规则体,这是一个搜索问题。形式化地说,给定正例集合与反例集合,学习任务是基于候选文字集合 $F = \{fk\}$ 来生成最优规则 r 。在命题规则学习中,候选文字是形如“ R (属性 i , 属性值 ij)”的布尔表达式,其中属性 i 表示样例第 i 个属性,属性值 ij 表示属性 i 的第 j 个候选值, $R(x, y)$ 则是判断 x, y 是否满足关系 R 的二元布尔函数。

最简单的做法是从空规则“+? + $\leftarrow$ ”开始,将正例类别作为规则头,再逐个遍历训练集中的每个属性及取值,尝试将其作为逻辑文字增加到规则体中,

若能使当前规则体仅覆盖正例,则由此产生一条规则,然后去除已被覆盖的正例并基于剩余样本尝试生成下一条规则。

以西瓜数据集 2.0 训练集为例,首先根据第 1 个样例生成文字“好瓜”和“色泽=青绿”加入规则,得到

$$\text{好瓜} \leftarrow (\text{色泽} = \text{青绿})$$

这条规则覆盖样例 1、6、10 和 17,其中有两个正例和两个反例,不符合“当前规则仅覆盖正例”的条件。于是,我们尝试将该命题替换为基于属性“色泽”形成的其他原子命题,例如“色泽=乌黑”;然而在这个数据集上,这样的操作不能产生符合条件的规则。于是我们回到“色泽=青绿”,尝试增加一个基于其他属性的原子命题,例如“根蒂=蜷缩”

$$\text{好瓜} \leftarrow (\text{色泽} = \text{青绿}) \wedge (\text{根蒂} = \text{稍蜷})$$

该规则仍覆盖了反例 17。于是我们将第二个命题替换为基于该属性形成的其他原子命题,例如“根蒂=稍蜷”:

$$\text{好瓜} \leftarrow (\text{色泽} = \text{青绿}) \wedge (\text{根蒂} = \text{稍蜷})$$

这条规则不覆盖任何反例,虽然它仅覆盖一个正例,但已满足“当前规则仅覆盖正例”的条件。因此我们保留这条规则并去除它覆盖的样例 6,然后将剩下的 9 个样例用作训练集。如此继续,我们将得到:

规则 1: $\text{好瓜} \leftarrow (\text{色泽} = \text{青绿}) \wedge (\text{根蒂} = \text{稍蜷});$

规则 2: $\text{好瓜} \leftarrow (\text{色泽} = \text{青绿}) \wedge (\text{敲声} = \text{浊响});$

规则 3: $\text{好瓜} \leftarrow (\text{色泽} = \text{乌黑}) \wedge (\text{根蒂} = \text{蜷缩});$

规则 4: $\text{好瓜} \leftarrow (\text{色泽} = \text{乌黑}) \wedge (\text{纹理} = \text{稍糊}).$

这个规则集覆盖了所有正例,未覆盖任何反例,这就是序贯覆盖法学得的结果。

上面这种基于穷尽搜索的做法在属性和候选值较多时会由于组合爆炸而不可行。现实任务中一般有两种策略来产生规则:第一种是“自顶向下”(top-down),即从比较一般的规则开始,逐渐添加新文字以缩小规则覆盖范围,直到满足预定条件为止;亦称为“生成一测试”(generate-then-test)法,是规则逐渐“特化”(specialization)的过程。第二种策略是“自底向上”(bottom-up),即从比较特殊的规则开始,逐渐删除文字以扩大规则覆盖范围,直到满足条件为

止;亦称为“数据驱动”(data-driven)法,是规则逐渐“泛化”(generalization)的过程。第一种策略是覆盖范围从大往小搜索规则,第二种策略则相反;前者通常更容易产生泛化性能较好的规则,而后者则更适合于训练样本较少的情形,此外,前者对噪声的鲁棒性比后者要强得多。因此,在命题规则学习中通常使用第一种策略,而第二种策略在一阶规则学习这类假设空间非常复杂的任务上使用较多。

下面以西瓜数据集 2.0 训练集为例来展示自顶向下的规则生成方法。首先从空规则“好瓜 $\leftarrow$ 开始”,逐一将“属性=取值”作为原子命题加入空规则进行考察。假定基于训练集准确率来评估规则的优劣, n/m 表示加入某命题后新规则在训练集上的准确率,其中 m 为覆盖的样例总数, n 为覆盖的正例数。如图 5-6 所示,经过第一轮评估,“色泽=乌黑”和“脐部=凹陷”都达到了最高准确率 $3/4$ 。

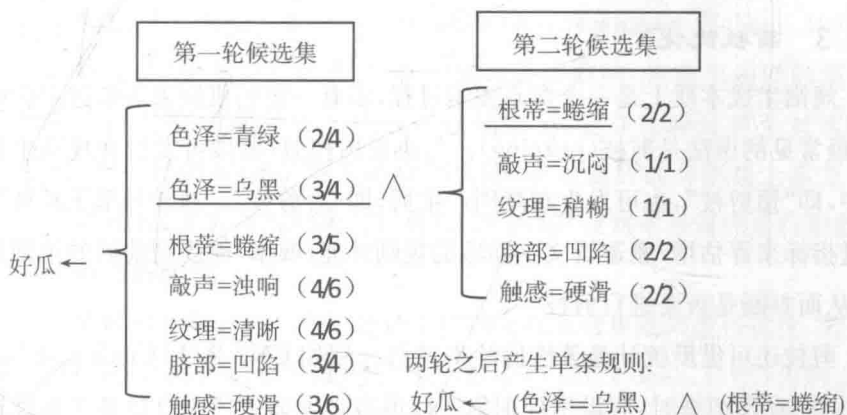


图 5-6 在西瓜数据集 2.0 训练集上“自顶向下”生成单条规则

将属性次序最靠前的逻辑文字“色泽=乌黑”加入空规则,得到

$$\text{好瓜} \leftarrow (\text{色泽} = \text{乌黑})$$

然后,对上面这条规则覆盖的样例,通过第二轮评估可发现,将图 5-7 中的五个逻辑文字加入规则后都能达到 100% 准确率,我们将覆盖样例最多且属性次序最靠前的逻辑文字“根蒂=蜷缩”加入规则,于是得到结果

$$\text{好瓜} \leftarrow (\text{色泽} = \text{乌黑}) \wedge (\text{根蒂} = \text{蜷缩})$$

规则生成过程中涉及一个评估规则优劣的标准,在上面的例子中使用的标

准是:先考虑规则准确率,准确率相同时考虑覆盖样例数,再相同时考虑属性次序。现实应用中可根据具体任务情况设计适当的标准。

此外,在上面的例子中每次仅考虑一个“最优”文字,这通常过于贪心,易陷入局部最优。为缓解这个问题,可采用一些相对温和的做法,例如采用“集束搜索”(beamsearch),即每轮保留最优的 b 个逻辑文字,在下一轮均用于构建候选集,再把候选集中最优的 b 个留待再下一轮使用。图 5-7 中若采用 $b=2$ 的集束搜索,则第一轮将保留准确率为 $3/4$ 的两个逻辑文字,在第二轮评估后就能获得下面这条规则,其准确率仍为 100% ,但是覆盖了 3 个正例:

好瓜 $\leftarrow (\text{脐部} = \text{凹陷}) \wedge (\text{根蒂} = \text{蜷缩})$

由于序贯覆盖法简单有效,几乎所有规则学习算法都以它为基础框架。它能方便地推广到多分类问题上,只需将每类分别处理即可;当学习关于第 c 类的规则时,将所有属于类别 c 的样本作为正例,其他类别的样本作为反例。

5.4.3 剪枝优化

规则生成本质上是一个贪心搜索过程,需有一定的机制来缓解过拟合的风险,最常见的做法是剪枝(pruning)。与决策树相似,剪枝可发生在规则生长过程中,即“预剪枝”,也可发生在规则产生后,即“后剪枝”。通常是基于某种性能度量指标来评估增/删逻辑文字前后的规则性能,或增/删规则前后的规则集性能,从而判断是否要进行剪枝。

剪枝还可借助统计显著性检验来进行。例如 CN2 算法(Clark and Niblett, 1989)在预剪枝时,假设用规则集进行预测必须显著优于直接基于训练样例集后验概率分布进行预测。为便于计算,CN2 使用了似然率统计量(Likelihood Ratio Statistics, LRS)。令 m_+ , m_- 分别表示训练样例集中的正、反例数目, $\hat{m}_+$, $\hat{m}_-$ 分别表示规则(集)覆盖的正、反例数目,则有

$$\text{LRS} = 2 \cdot \left(\hat{m}_+ \log_2 \frac{\left(\frac{\hat{m}_+}{\hat{m}_+ + \hat{m}_-} \right)}{\left(\frac{m_+}{m_+ + m_-} \right)} + \hat{m}_- \log_2 \frac{\left(\frac{\hat{m}_-}{\hat{m}_+ + \hat{m}_-} \right)}{\left(\frac{m_-}{m_+ + m_-} \right)} \right) \quad (5-15)$$

这实际上是一种信息量指标,衡量了规则(集)覆盖样例的分布与训练集经

验分布的差别:LRS越大,说明采用规则(集)进行预测与直接使用训练集正、反例比率进行猜测的差别越大;LRS越小,说明规则(集)的效果越可能仅是偶然现象。在数据量比较大的现实任务中,通常设置为在LRS很大(例如0.99)时CN2算法才停止规则(集)生长。

后剪枝最常用的策略是“减错剪枝”(Reduced Error Pruning, REP)(Brunkand Pazzani, 1991),其基本做法是:将样例集划分为训练集和验证集,从训练集上学得规则集 n 后进行多轮剪枝,在每一轮穷举所有可能的剪枝操作,包括删除规则中某个文字、删除规则结尾文字、删除规则尾部多个文字、删除整条规则等,然后用验证集对剪枝产生的所有候选规则集进行评估,保留最好的那个规则集进行下一轮剪枝,如此继续,直到无法通过剪枝提高验证集上的性能为止。

REP剪枝通常很有效(BrunkandPazzani, 1991),但其复杂度是 $O(m^4)$, m 为训练样例数目。IREP(Incremental REP)(Fürrnkranzand Widmer, 1994)将复杂度降到 $O(m \log_2 m)$,其做法是:在生成每条规则前,先将当前样例集划分为训练集和验证集,在训练集上生成一条规则 r' ,立即在验证集上对其进行REP剪枝,得到规则 r' ;将 r' 覆盖的样例去除,在更新后的样例集上重复上述过程。显然,REP是针对规则集进行剪枝,而IREP仅对单条规则进行剪枝,因此后者比前者更高效。

若将剪枝机制与其他一些后处理手段结合起来对规则集进行优化,则往往能获得更好的效果。以著名的规则学习算法RIPPER(Cohen, 1995)为例,其泛化性能超过很多决策树算法,而且学习速度也比大多数决策树算法更快,奥妙就在于将剪枝与后处理优化相结合。

RIPPER算法描述如图5-7所示。它先使用IREP*剪枝机制生成规则集 R 。IREP*(Cohen, 1995)是IREP的改进,主要是以 $\frac{\hat{m}_+ + (m_- - \hat{m}_-)}{m_+ + m_-}$ 取代了IREP使用的准确率作为规则性能度量指标,在剪枝时删除规则尾部的多个文字,并在最终得到规则集之后再进行一次IREP剪枝。RIPPER中的后处理机制是为了在剪枝的基础上进一步提升性能。对 R 中的每条规则 r_i ,RIPPER为它产生两个变体。

ri' : 基于 ri 覆盖的样例, 用 IREP * 重新生成一条规则 ri' , 该规则称为替换规则(replacement rule)。

ri'' : 对 ri 增加文字进行特化, 然后再用 IREP * 剪枝生成一条规则 ri'' , 该规则称为修订规则(revised rule)。

接下来, 把 ri' 和 ri'' 分别与 R 中除 ri' 之外的规则放在一起, 组成规则集 R' 和 R'' , 将它们与 R 一起进行比较, 选择最优的规则集保留下来。这就是图 5-7 中算法第 4 行所做的操作。

为什么 RIPPER 的优化策略会有效呢? 原因很简单: 最初生成 R 的时候, 规则是按序生成的, 每条规则都没有对其后产生的规则加以考虑, 这样的贪心算法本质常导致算法陷入局部最优; RIPPER 的后处理优化过程将 R 中的所有规则放在一起重新加以优化, 恰是通过全局的考虑来缓解贪心算法的局部性, 从而往往能得到更好的效果(Fürnkranz et al, 2012)。

输入: 训练样例集 D ;

重复次数 k ;

训练轮数 T ;

过程:

1: $R = \text{IREP} * (D)$;

2: $i = 0$;

3: repeat

4: $R' = \text{PostOpt}(R)$

5: $D_i = \text{NotCovered}(R', D)$

6: $R_i = \text{IREP} * (D_i)$;

7: $R = R' \cup R_i$

8: $i = i + 1$

9: until $i = k$

输出: 规则集 R

图 5-7 RIPPER 算法

5.4.4 阅读材料

规则学习是“符号主义学习”(symbolism learning)的主要代表,是最早开始研究的机器学习技术之一(Michalski,1983)。(Fürnkranzetal.,2012)对规则学习做了比较全面的总结。

序贯覆盖是规则学习的基本框架,最早在(Michalski,1969)的AQ中被提出,AQ后来发展成一个算法族,其中比较著名的有AQ15(Michalskietal.,1986)、AQ17-HCI(Wnekand Michalski,1994)等。受计算能力的制约,早期AQ在学习时只能随机挑选一对正反例作为种子开始训练,样例选择的随机性导致AQ学习效果不稳定。PRISM(Cendrowska,1987)解决了这个问题,该算法最早采用自顶向下搜索,并显示出规则学习与决策树学习相比的优点:决策树试图将样本空间划分为不重叠的等价类,而规则学习并不强求这一点,因此后者学得模型能有更低的复杂度。虽然PRISM的性能不如AQ,因此在当时反响不大,但今天来看,它是规则学习领域发展的重要一步。

CN2(Clarkand Niblett,1989)采用集束搜索,是最早考虑过拟合问题的规则学习算法。(Fürnkranz,1994)显示出后剪枝在缓解规则学习过拟合中的优势。RIPPER[Cohen,1995]是命题规则学习技术的高峰,它融合了该领域的许多技巧,使规则学习在与决策树学习的长期竞争中首次占据上风,作者主页上的C语言RIPPER版本至今仍代表着命题规则学习的最高水平。

关系学习的研究一般认为始于Winston(1970);由于命题规则学习很难完成此类任务,一阶规则学习开始得以发展。FOIL通过变量替换等操作把命题规则学习转化为一阶规则学习,该技术至今仍有使用,例如2010年卡耐基梅隆大学开展的“永动语言学习”(Never-Ending Language Learning,NELL)计划即采用FOIL来学习自然语言中的语义关系(Carlsonetal,2010)。

Muggleton(1991)提出了“归纳逻辑程序设计”(ILP)这个术语,在GOLEM(Muggletonand Feng,1990)中克服了许多从命题逻辑过渡到一阶逻辑学习的困难,并确立了自底向上归纳的ILP框架。最小一般泛化(LGG)最早由(Plotkin,1970)提出,GOLEM则使用了RLGG。PROGOL(Muggleton,1995)将逆归结改进为逆蕴含(inverse entailment)并取得了更好效果。新谓词发明方

面近年有一些新进展(Muggleton and Lin, 2013)。由于 ILP 学得的规则几乎能直接被 PROLOG 等逻辑程序解释器调用,而 PROLOG 在专家系统中常被使用,因此 ILP 成为连接机器学习与知识工程的重要桥梁。PROGOL(Muggleton, 1995)和 ALEPH(Srinivasan, 1999)是应用广泛的 ILP 系统,其基本思想已在本章关于 ILP 的部分有所体现。Datalog(Cerietal, 1989)则对数据库领域产生了很大影响,甚至影响了 SQL1999 标准和 IBMDB2。ILP 方面的重要读物有 Muggleton(1992)、Lavrac and Dzeroski(1993),并且有专门的国际归纳逻辑程序设计会议(ILP)。ILP 复杂度很高,虽在生物数据挖掘和自然语言处理等任务中取得一些成功(Bratko and Muggleton, 1995),但问题规模稍大就难以处理。因此,这方面的研究在统计学习兴起后受到一定抑制。近年来随着机器学习技术进入更多应用领域,在富含结构信息和领域知识的任务中,逻辑表达的重要性逐渐凸显出来,因此出现了一些将规则学习与统计学习相结合的努力,例如试图在归纳逻辑程序设计中引入概率模型的“概率归纳逻辑程序设计”(probabilistic ILP)(DeRaedt et al, 2000),给贝叶斯网中的节点赋予逻辑意义的“关系贝叶斯网”(relational Bayesian network)(Jaeger, 2002)等。事实上,将关系学习与统计学习相结合是机器学习发展的一大趋势,而概率归纳逻辑程序设计是其中的重要代表,其他重要代表还有概率关系模型(Riedman et al, 1999),贝叶斯逻辑程序(Bayesian Logic Program)(Kersting et al, 2000)、马尔可夫逻辑网(Markov logic network)(Richardson and Domingos, 2006)等,统称为“统计关系学习”(statistical relational learning)(Getoor and Taskar, 2007)。

5.5 基于支持向量机的强化学习

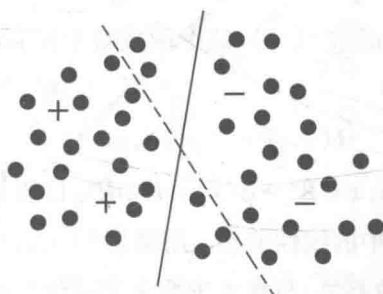
5.5.1 半监督 SVM

SVM 是一种典型的核机器学习方法,是借助于最优化方法解决机器学习问题的新工具,由统计学习理论发展而来。最初于 20 世纪 90 年代由 Vapnik 提出,近年来在其理论研究和算法实现方面都取得了突破性进展,开始成为克服维数灾难、过学习及局部最优等传统困难的有效手段。

半监督支持向量机(Semi-Supervised Support Vector Machine, S3VM)是

支持向量机在半监督学习上的推广。在不考虑未标记样本时,支持向量机试图找到最大间隔划分超平面,而在考虑未标记样本后,S3VM 试图找到能将两类有标记样本分开,且穿过数据低密度区域的划分超平面,如图 5-8 所示,这里的基本假设是“低密度分隔”(low-density separation),显然,这是聚类假设在考虑了线性超平面划分后的推广。

S3VM 划分超平面



SVM 划分超平面

图 5-8 半监督支持向量机与低密度分隔

(“+”“-”分别表示有标记的正、反例,灰色点表示未标记样本)

半监督支持向量机中最著名的是 TSVM (Transductive Support Vector Machine) (Joachims, 1999)。与标准 SVM 一样,TSVM 也是针对二分类问题的学习方法。TSVM 试图考虑对未标记样本进行各种可能的标记指派 (label assignment),即尝试将每个未标记样本分别作为正例或反例,然后在所有这些结果中,寻求一个在所有样本(包括有标记样本和进行了标记指派的未标记样本)上间隔最大化的划分超平面。一旦划分超平面得以确定,未标记样本的最终标记指派就是其预测结果。

5.5.2 核学习

核机器学习方法是基于统计学习理论和核技术建立的,以 SVM 为其核心算法的一类统计机器学习算法。该方法采取的基本策略是将数据嵌入到一个可以发现线性关系的空间,可用图 5-9 描述。这里可以用模块化的方式描述:首先为初始映射,即输入空间 X 中的数据由核函数隐式映射到特征空间 F ,然后在该特征空间中构造最优超平面,设计一个线性算法或利用已有的线性算

法,将其转化为仅涉及内积运算的优化问题。

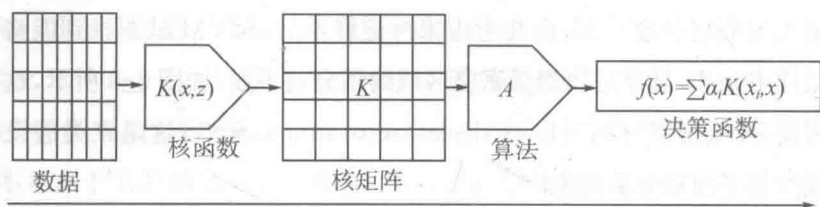


图 5-9 核方法的应用阶段

核(Kernel)为一个函数 $k(\cdot)$, 这个函数对于所有的 $x, z \in X$, 满足式(5-16)所述条件

$$k(x, z) = \langle \varphi(x), \varphi(z) \rangle \quad (5-16)$$

式中, φ 为嵌入的映射 $\varphi: x \in \mathbf{R}^n \rightarrow \varphi(x) \in F \subseteq \mathbf{R}^n$, 目的是把低维空间中的非线性关系转化为高维空间中的线性关系。用满足 Mercer 定理的核函数来替代内积运算, 导出只与样本数有关, 与样本维数无关的优化问题。这种思想使得利用指数维数甚至无限维数的特征空间成为可能。

核机器学习方法主要可分为两类: 有监督核机器学习方法和无监督核机器学习方法。目前, 有监督核机器学习主要包括支持向量机、核 Fisher 判别式, 无监督核机器学习主要为核 PCA。

5.5.3 SVM

SVM 实现结构风险最小化即 SRM 准则有两种思路: 一是在函数集的每个子集中求最小经验风险, 然后选择使最小经验风险和置信范围之和最小的子集, 这种方法比较费时, 当子集数目很大甚至是无穷时是不可行的; 二是设计函数集的某种结构使每个子集中都能取得最小的经验风险(如使训练误差为 0), 然后只需选择适当的子集使置信范围最小, 则这个子集中使经验风险最小的函数就是最优函数, SVM 实际上就是这种思想的具体实现。

SVM 集成了统计学习理论的 VC 维和结构风险最小原理, 该方法用少数支持向量代表整个样本集, 其原理主要包括分类和回归两大部分。SVM 从线性可分情况下的最优分类面发展而来, 基本思想可用图 5-10 的两维情况说明。图中, 实心点和空心点分别代表两类线性可分样本, H 为分类线, H_1 和 H_2 分别为过两类中离分类线最近的样本且与分类线平行的直线, 它们之间的距离叫

作分类间隔(margin),所谓最优分类线为既能正确分类两类样本,又可使分类间隔最大的分类线。 H_1 和 H_2 上的点为支持向量,即对应于 Lagrange 乘子不为零的点。

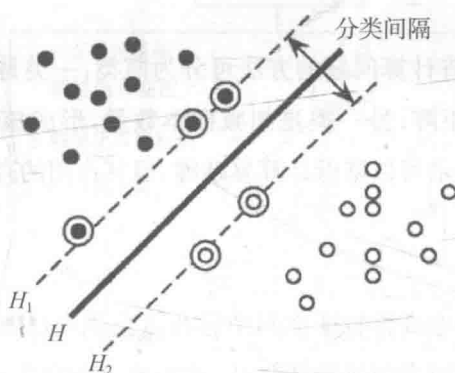


图 5-10 SVM 基本思想

5.5.4 SVM 的重要概念

SVM 为核函数引入机器学习的首例成功运用,我们结合如图 5-9 所示的核方法应用的各个阶段全面阐述 SVM 的几个重要概念。

1. 核函数

在 SVM 中,需要选择核函数 $k(\cdot)$,或者说需要选择一个映射 $\varphi(\cdot)$,把输入变量所在的空间 X 映射到另一个空间 F ,这个空间 F 可以是有限维空间,也可以是无限维空间。一般来说它是一个 Hilbert 空间,也称特征空间。选择不同的核函数,或不同的映射及其相应的 Hilbert 空间,相当于选择不同的内积。这意味着以不同的标准对相似性和相似程度进行估价。在解决实际问题中,映射的选择是非常重要的。映射选定以后,立即可由其内积构造出核函数,从而使用 SVM 进行计算。

在使用 SVM 时,上述核函数起着直接的作用,实际应用中甚至不需要知道具体的映射是什么,只要选定核函数就可以。 $k(\cdot)$ 有较大的选择范围,但必须满足 Mercer 定理。常见的几种核函数有线性核、多项式核、高斯径向基核、Sigmoid 核以及傅里叶核。

2. 核矩阵

核矩阵是核方法理论中的核心成分,是核机器实现方面的中心数据结构。它包含为了执行学习步骤所有能够获得的信息,唯一的例外是它不包含监督学习情况下的输出标签。核矩阵是核算法的信息瓶颈,其性质直接影响到学习系统的各个环节。

目前,解决核矩阵计算问题的方法可分为两类:一类是将核矩阵某些数据用零替换,形成稀疏矩阵;另一类是削减样本数量,形成核矩阵的低秩逼近矩阵。核矩阵的稀疏表示可以降低计算复杂度,但其占用的存储空间并没有明显减少。

3. 损失函数

核机器学习就是在高维特征空间中寻找适当规则以解决低维空间中不易处理的问题。要确定对期望风险的估计,首先要确定损失函数,常见的损失函数为 ϵ^- 不敏感损失函数

$$c(\epsilon) = \max(0, |\epsilon| - \epsilon), \epsilon > 0 \quad (5-17)$$

该函数可导致解的稀疏性。此类函数常应用于回归问题中。在选择 ϵ 值的大小时,要考虑样本中噪声的影响。目前, ϵ^- 不敏感损失函数主要包括线性不敏感损失函数、二次不敏感损失函数、Huber 损失函数等。

4. 训练算法

标准支持向量机的训练过程为凸二次规划(Quadratic Programming, QP)过程,可用传统的标准二次型优化技术来解决,为减少 SVM 学习算法的计算复杂性,利用 SVM 优化问题本身所具有的特性,近年来提出了几种改进的支持向量机的训练算法。主要有三大类:以 SMO 为代表的分解算法、以 EG 为代表的多变量更新算法、以 IDA 为代表的序列方法。另外,还有多种 SVM 的变形算法。

5.6 基于 SVM 的强化学习

5.6.1 基于 SVM 的 Q 学习结构

基于 SVM 的连续空间 Q 学习方法的的思想是,将连续状态空间下的 Q 学习构建为 SVM 的回归估计问题,利用 SVM 良好的泛化以及非线性逼近性能

进行训练集的更新,时间窗保持不变;若违反,则原有的 SVM 模型拟合该数据时会产生比较大的误差,因此,应滚动时间窗,重新训练得到新的 SVM 模型。

为了构造在线 SVM,样本是窗式移动的。设 t 时刻 SVM 学习训练样本集由过去 l 组数据构成 $D = \{(x_i, Q_i) \mid i = t-l, t-l+1, \dots, t-1\}$, 其中样本输入数据 $x_{t-1} = (s_{t-1}, a_{t-1})^T \in \mathbf{R}^{n+1}$ 表示由 $t-1$ 时刻 n 维系统状态 $s_{t-1} = (s_1(t-1), s_2(t-1), \dots, s_n(t-1))^T \in \mathbf{R}^{n+1}$ 和 1 维动作 $a_{t-1} \in \mathbf{R}$ 构成的状态—动作对,样本输出数据 $Q_{t-1} \in \mathbf{R}$ 为 $(t-1)$ 时刻 Q 学习系统的 Q 值。为了解决 Q 学习过程中探索和利用的两难问题, SVM 的输出 Q 值要被送入随机动作选择器,此处采用近似贪心且连续可微的 Boltzmann-Gibbs 分布作为动作选择策略。

5.6.2 基于滚动时间窗机制的 SVM

为了充分利用数据的信息,对于 l 区间内的样本,应当根据数据采样时刻的不同,给予不同程度的考虑,当采样时刻离当前时刻越远,则其在建模时所占比重将减少,离当前时刻越近的数据,则在建模时给予更多的考虑。因此,采用线性遗忘方式对时间窗内的数据分配不同的权值注 γ_i ,以反映区间内样本的重要程度。

$$h_i = \frac{2(i-t+l+1)}{l(l+1)}, \sum_{i=t-l}^{t-1} \gamma_i = 1 \quad (5-18)$$

在回归估计中,为保持 SVM 算法的稀疏性,去除原变量空间中大部分误差较小的样本点,引入 ϵ -不敏感损失函数

$$c(x, Q, f(x)) = |Q - f(x)|_\epsilon = \max\{0, |Q - f(x)| - \epsilon\} \quad (5-19)$$

如图 5-11 所示, SVM 形式上类似于一个神经网络,输出是中间节点的线性组合,因此,待求回归函数的形式为

$$Q = f(x) = \omega^T \varphi(x) + b \quad (5-20)$$

式中, ω 为权重向量, b 为偏置项,可以通过求解凸最优化问题得到系数 ω 和 b 。引入松弛变量 $\epsilon^{(*)} = (\epsilon_{t-l}, \epsilon_{t-l}^*, \dots, \epsilon_{t-1}, \epsilon_{t-1}^*)^T$ 和正则化参数 C , 得到 ϵ -支持向量机的原始优化问题

$$\omega \in \mathbf{R}^{n+1}, \epsilon^{(*)} \in \mathbf{R}^{2l}, b \in \mathbf{R}^{F_D(\omega, \epsilon^{(*)})} = \frac{1}{2} \omega^2 + C \sum_{i=t-l}^{t-1} (\epsilon_i + \epsilon_i^*) \quad (5-21)$$

式中, $(*)$ 是表示向量有 $*$ 和无 $*$ 两种情况的简单记号, $\varphi(x)$ 将变量从输入空

间非线性地映射到高维特征空间,从而将非线性关系转化为线性关系。在低维输入空间中引入如式(5-22)描述的径向基核函数对应高维特征空间的内积运算。

$$k(x_i, x_j) = \varphi(x_i) \cdot \varphi(x_j) = \exp\left(-\frac{(x_i - x_j)^2}{2\sigma^2}\right),$$

$$j = t-l, t-l+1, \dots, t-1 \quad (5-22)$$

式(5-21)中的正则化参数 C 和式(5-22)中的核宽度 σ 是非常重要的参数,这些参数的选择可以根据经验、自举法、交叉验证和从统计学习理论导出 VC 维的界等方法进行确定,其中最常用的方法是交叉验证法。

引入 Lagrange 乘子 $\alpha^{(*)} = (\alpha_{t-l}, \alpha_{t-l}^*, \dots, \alpha_{t-1}, \alpha_{t-1}^*)^T$, 利用 Lagrange 乘子法构造原始问题的对偶问题并求解下述最优化问题即可得到最优解 $\alpha^{(*)}$ 。

$$s. t. \quad \sum_{i=t-l}^{t-1} (\alpha_i^* - \alpha_i) = 0$$

$$0 \leq \alpha_i, \alpha_i^* \leq \frac{C}{l} \quad (5-23)$$

已经证明:原始问题式(5-21)关于 ω 的唯一解可用对偶问题式(5-23)的解来表示,即

$$\omega = \sum_{i=t-l}^{t-1} (\alpha_i^* - \alpha_i) \varphi(x_i) \quad (5-24)$$

因此,最优回归估计函数可表示为

$$f(x) = \sum_{i=t-l}^{t-1} \omega_i K(x_i, x) + b \quad (5-25)$$

式中, $\omega_i = \alpha_i^* - \alpha_i$ 。

$W_D(\alpha^{(*)})$ 分别对 α_i 和 α_i^* 求一阶偏导,可得到回归分析中的 KKT 条件如下:

$$\alpha_i^{(*)} = 0 \Rightarrow |Q_i - f(x_i)| \leq \epsilon$$

$$0 < \alpha_i^{(*)} < C \Rightarrow |Q_i - f(x_i)| = \epsilon$$

$$\alpha_i^* = C \Rightarrow |Q_i - f(x_i)| \geq \epsilon \quad (5-26)$$

由上述 KKT 条件可知,利用任意一个支持向量就可以计算得到 b 的值。为得到更为平滑的回归曲线,取所有支持向量计算得到 b 的平均值 $\bar{b}$, 即

$$b = \bar{b} = \delta_{SV} + Q_{SV} - \sum_{i=t-l}^{t-1} (\alpha_i^* - \alpha_i) K(x_i, x_{SV}) \quad (5-27)$$

式中, $\delta_{SV} = f(x_{SV}) - Q_{SV}$ 为支持向量 x_{SV} 的估计误差。

5.6.3 算法步骤

根据上述分析,给出基于 SVM 的 Q 学习方法的计算步骤如下。

Step1:初始化 Q 学习控制器以及 SVM 回归模型的参数。

Step2:检测系统当前状态 s_t 。

Step3:构造 t 时刻 SVM 的学习训练样本集 D ,由最优化问题(5-23)得到最优解 $a^{(*)}$,由式(5-24)和式(5-27)得到 SVM 的结构参数 ω 和 b 。

Step4:由式(8-25)计算得到待选动作对应的 Q 值,根据贪心策略选择最大 Q 值对应的动作 a_t 。

Step5:执行动作 a_t ,获取下一时刻状态 s_{t+1} 及立即回报 r_t 。

Step6:按照 Q 值速归学习式更新 Q 值,得到目标值 Q 。

Step7:判断新数据 (x_t, Q_t) 是否违反 KKT 条件,若不违反,则保持时间窗不变,若违反,则将该数据加入训练集并滚动时间窗。

Step8:若不满足学习结束条件, $t \leftarrow t+1$, 转 Step2。

5.6.4 仿真研究

分别采用蒋国飞给出的基于 BP 神经网络的 Q 学习方法与本文所提 Q 学习方法进行 30 回合的独立仿真运行。仿真过程中,基于 SVM 的 Q 学习控制参数为: $\gamma=0.95$, $\eta=0.15$, $C=100$;用于动作选择策略的相关参数为:退火因子 $\beta=0.95$,温度上限值 $T_{\max}=0.1$,温度下限值 $T_{\min}=0.001$, σ 在 $[0.3, 0.5]$ 区间内随机取值 0.005, $l=50$,采样周期 $T_s=0.02s$;基于 BP 网络的 Q 学习控制参数为: $\gamma=0.95$, $\eta=0.15$, $\beta=0.95$, $T_{\max}=0.1$, $T_{\min}=0.001$, $T_s=0.02s$,网络拓扑结构为 5-7-1,网络隐层和输出层的激活函数分别为 Sigmoid 型和线性函数,网络学习率为 0.2。若在某次学习中,学习系统能够在 2000 时间步内保持系统状态在设定范围内,则认为学习系统达到本次仿真的要求。

在 30 回合的独立仿真运行中,基于 BP 网络的 Q 学习系统仅有 4 个回合能够成功平衡倒立摆达 2000 时间步,而基于 SVM 的 Q 学习系统在每次仿真运行中均能以较少的尝试次数获得倒立摆平衡控制策略,表明基于 SVM 的 Q 学习系统具有较高的学习效率和泛化性能。图 5-12(a)和图 5-12(b)分别给出

了基于 SVM 和基于 BP 网络的 Q 学习系统的一次典型运行的学习曲线,横坐标为尝试次数,纵坐标为每次学习实验中倒立摆保持平衡的成功次数。从图中可以看出,基于 SVM 的 Q 学习系统在经过大约 48 次学习后,能够达到预定目标,而基于 BP 网络的 Q 学习系统则需经过 1041 次学习后才可以将摆杆平衡 2000 时间步。图 5-12(c)和图 5-12(d)给出了两种 Q 学习控制器在分别经过 48 次和 1041 次学习后,系统的有关状态变化数据。仿真结果表明在有限的尝试次数内,本节所提控制算法可以达到预定的控制目标,验证了 SVM 与 Q 学习相结合的可行性。

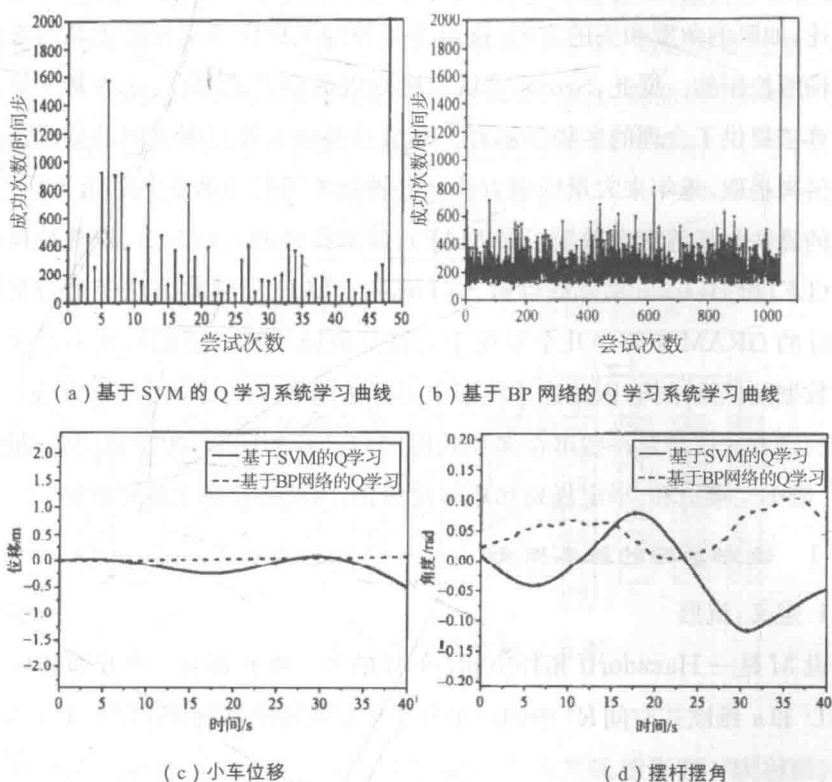


图 5-12 倒立摆平衡控制仿真结果

本节提出了一种基于 SVM 的 Q 学习方法,该方法的特点主要体现在如下三个方面。

(1) SVM 能够对 Q 学习的状态—动作对的 Q 值进行在线估计,解决连续状态空间泛化中易出现的“维数灾难”问题,可以有效解决具有连续状态的非线

性系统的强化学习控制。

(2) 基于滚动时间窗机制的 SVM 能够实现在线学习, 并且比常规 SVM 具有较快的估计速度。

(3) 对系统实时检测得到的新数据进行 KKT 条件判断, 可以保证随着每次时间窗的变动, 模型总能朝着更为准确的方向改进。

5.7 流形学习与图谱学习

2000 年, Seung 等发现整个细胞群的触发率可以由少量变量组成的函数来描述, 如眼的角度和头的方向, 这隐含了神经元群体活动性是由其内在的低维结构所控制的。据此, Seung 等认为感知以流形方式存在, 这为基于流形的学习算法提供了合理的生物学解释。那么这种嵌入在高维空间的低维流形结构如何来提取, 近年来大量的谱方法已经被用来进行维数简约和流形学习, 代表性的算法包括等度规映射 (ISOMAP)、局部线性嵌入 (LLE)、拉普拉斯特征映射 (LE) 和 Hessian 局部线性嵌入 (Hessian LLE)。这些方法能通过提取特别设计的 GRAM 矩阵中几个对应于大特征值或小特征值的特征向量来揭示高维数据的低维结构, 因此这些流形学习算法也称为流形学习的谱方法。流形学习的谱方法仅涉及一些可在多项式时间内计算的问题, 如特征问题、最短路问题、最小二乘拟和、半定规划和矩阵对角化, 因而在计算上是可行的。

5.7.1 流形学习的基本概念

1. 定义: 流形

设 M 是一 Hausdorff 拓扑空间, 若 M 的每一点 p 都有一个开邻域 $U \subset M$, 使得 U 和 n 维欧式空间 R^n 中的一个开子集是同胚的, 则称 M 是一个 n 维拓扑流形, 简称为 n 维流形。

2. 定义: 坐标卡

假定在定义“流形”中所提到的同胚是 $\varphi: U \rightarrow \varphi(U) \subset R^n$, 其中, $\varphi(U)$ 是 R^n 中的开集, 则称 (U, φ) 为流形 M 的一个坐标卡。

3. 定义: 维数约简

数据矩阵 $X_{n \times D}$ 由 n 个 D 维的数据向量 x_i 组成, 而该数据集的本征维数为

$d(d < D, \text{一般情况 } d \leq D)$ 。在数学中,本征维数是指嵌入到 D 维高维空间的数据集 X 接近或依附的流形的维数 d 。维数约简方法就是把高维数据集 X 转化为 d 维的数据集 Y ,同时尽可能地保持原高维数据的几何结构信息。总之,嵌入的流形的结构以及数据集 X 的本征维数都是来知的。因此,降维是一种不适定问题,只有假定了数据特定的性质(如本征维数)才能够求解。

5.7.2 流形学习的降维方法分类

我们根据各种流形学习降维方法的思想实质,将其分成两大类:第一类是构建关系矩阵的方法,主要包括等度规映射、局部线性嵌入、随机近邻嵌入、漫散射、Laplace 特征映射、最大方差展开、等角特征映射;第二类是基于局部模型的全局坐标对齐方法,包括 Hessian 局部线性嵌入、局部切空间排列、图册化流形、局部线性对齐、黎曼流形学习,分类情况如图 5-13 所示。

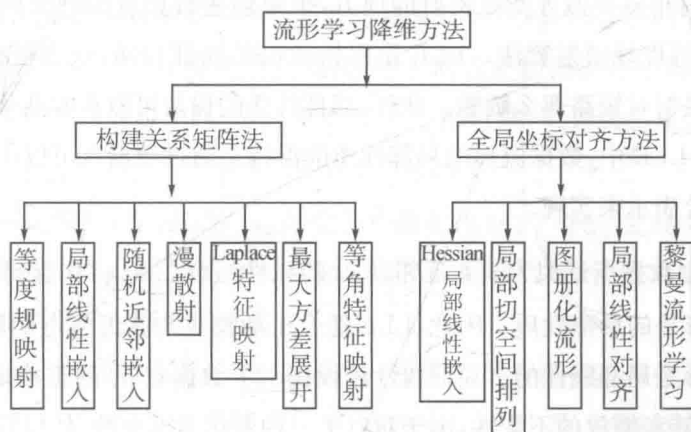


图 5-13 流形学习降维分类图

5.7.3 构建关系矩阵的方法

构建关系矩阵的方法关键就在于构建合适的关系矩阵,主要包括距离矩阵、相似矩阵、局部线性重构矩阵等,关系矩阵的构造成功与否对最终的降维结果影响很大,下面将详细讨论此类的七种算法:等度规映射、局部线性嵌入、随机近邻嵌入、Laplace 特征映射、漫散射、最大方差展开、等角特征映射。

1. 等度规映射(ISOMAP)

ISOMAP 是一种保持数据点成对测地线距离的方法。测地线距离是流形

上的两点沿流形曲面的最短距离。该方法的算法有以下三步:首先构建近邻图;然后应用 Dijkstra 最短路径算法求取图上成对近似测地线距离;最后应用多维尺度变换获得嵌入在高维空间中的流形低维坐标表示。

ISOMAP 算法一个比较大的缺陷是拓扑稳定性差。若要为近邻图 G 创建充足的连接,就可能出现“短路”现象,会严重影响其性能,现已提出了很多方法来克服“短路”问题,如移除路径连接中高密度区的数据点的一些连接,或移除近邻图中影响局部线性的最近邻点。此外,ISOMAP 算法还可能受到流形存在孔洞的影响,这一问题可应用容忍空洞的流形学习方法来解决,ISOMAP 算法的第三个缺点是要求流形必须是凸结构的。尽管 ISOMAP 算法有以上的缺点,但它还是有很多成功的应用,例如生物医学数据可视化等。

2. 局部线性嵌入(LLE)

LLE 有很多类似等度规映射的地方,也要创建数据点的近邻图的降维方法,相对于等度规映射算法。LLE 是保持数据的局部性质,这也使得 LLE 没有等度规映射对短路那么敏感。此外,局部性质的保持可以在非凸流形上成功地应用,在 LLE 中,数据流形的局部性质的保持是通过数据点可以用其近邻点的线性组合表示来实现。

LLE 把数据点近似为其 k 近邻点 x_{ij} 的线性组合 $\sum_{j=1}^k \omega_{ij} x_{ij}$ 来保持数据点 x_i 附近的流形是局部性质。因此,LLE 适合于数据点与其近邻点的超平面上,故假设流形是局部线性的。局部线性假设揭示了数据点 x_i 的重构矩阵 ω_i ,具有平移、旋转和缩放的不变性,由于对以上三种变化的不变性,任何到低维空间的超平面线性映射都保持低维空间重构的权重,换句话说,如果低维数据表示能够保持流形的局部几何结构,那么近似高维空间数据点的其近邻点线性组合矩阵 ω_i ,可以用低维数据点 y_i 的近邻点来重构低维数据 y_i ,所以,求解低维数据表示 Y 等价于最小化下面目标函数:

$$\varphi(Y) = \sum (y_i - \sum_{j=1}^k \omega_{ij} y_{ij})^2 \quad (5-28)$$

式(5-28)变化为

$$\varphi(Y) = (Y - WY)^2 = Y^T (I - W)^T (I - W) Y \quad (5-29)$$

由上式知,求解式(5-28)对应目标函数最小值的低维表示 y_i ,可由矩阵

$(I-W)^T(I-W)$ 的最小 d 个非零特征值对应的特征向量得到。其中, I 为 $n \times n$ 的单位矩阵, LLE 已经成功地应用于超分辨率分析和声源定位, 然而, 试验表明 LLE 也存在不足, 文献^①表明 LLE 算法不能成功地使简单人工生物医学数据库可视化。LLE 的一种拓展方法可以处理流形上包含很多洞的困难问题。

3. Laplace 特征映射(LE)

与 LLE 类似, LE 算法以保持流形上局部性质来求解低维数据表示。在该算法中, 局部性质是考虑近邻点之间的两两距离, 该算法由最小化低维空间中数据点与其近邻点的距离来计算数据的低维表示, 流形的结构可由选择适当权重的近邻图来近似, 在低维数据表示中, 数据点与其最近邻点的距离对目标函数的贡献比该数据点与其第二近邻数据的要大, 应用谱理论, 目标函数最小化转化为一个特征分解问题。

首先, 该算法要创建每一个数据点与其 k 近邻点构成的相邻图 G , 相邻图 G 的每个数据点 x_i 与 x_j 都由边来连接, 边上的权重由高斯核函数来计算, 进而得到一个稀疏的邻接矩阵 w , 对低维表示 y_j 的计算, 就是最小化目标函数

$$\varphi(Y) = \sum_{i,j} (y_i - y_j)^2 \omega_{ij} \quad (5-30)$$

在目标函数中, 大的权重 ω_{ij} 对应于小的数据点 x_i 与 x_j 之间的距离。因此, 低维表示 y_i 与 y_j 距离变化对目标函数的影响很大, 即使得高维空间中相邻的点映射到低维空间变得更近。

密度矩阵 M 和相邻图 G 的 Laplacian 矩阵 L 的计算使得目标函数最小化问题转化为一个特征问题, 邻接矩阵 W 的密度矩阵 M 是一个对角矩阵, 其对角元素是矩阵 W 的行元素之和, 图 Laplacian 矩阵 L 由式 $L = M - W$ 计算, 则式(5-30)变为

$$\varphi(Y) = \sum_{i,j} (y_i - y_j)^2 \omega_{ij} = 2Y^T L Y \quad (5-31)$$

因此, 最小化 $\varphi(Y)$ 等价于最小化 $Y^T L Y$, 低维的数据表示 Y 可通过广义特征问题来求解。

$$L v = \lambda M v \quad (5-32)$$

①LEEJA, VERLCYSEN M. Nonlinear dimensionality reduction of data manifolds with essential loops[J]. Neuro computing, 2005, 67: 29-53.

对应 d 个最小非零特征值的特征向量 v_i , 形成了低维数据表示 Y 。

4. 漫散射(DM)

DM 的结构来源于动力学系统领域, 该方法在数据图上定义了一个马尔科夫(Markov)随机游走。经过很多步随机游走的实施以后, 数据点间的相似度就可以度量出来, 这种度量方法被定义为 Diffusion 距离。在数据的低维描述中, 两两之间的 Diffusion 距离尽可能地保持不变。

DM 算法步骤: 首先, 要建立数据图结构。图的边上的权重由高斯核函数来计算, 得到矩阵 $W = \{\omega_{ij}\}$ 为

$$\omega_{ij} = e^{-\frac{x_i - x_j}{2\sigma^2}} \quad (5-33)$$

其中, σ 为高斯分布的标准差。其次, 规范化矩阵 W , 即使矩阵的每一列的和为 1, 得到的新的矩阵为 $P^{(1)}$:

$$P^{(1)} = \frac{\omega_{ij}}{\sum_k \omega_{ik}} \quad (5-34)$$

因为 DM 来源于动力学理论, 所以得到的矩阵 $P^{(1)}$ 可认为是马尔科夫矩阵, 在动力学中把它定义为动力学过程的前向转化概率矩阵。因此, 矩阵 $P^{(1)}$ 也表示为在一步迭代中的一个数据点向另一个数据点转化的概率。第 t 步的前向概率矩阵定义为 $(P^{(1)})^t$, 应用随机游走的前向概率 $P_{ij}^{(t)}$, Diffusion 距离定义为

$$D^{(t)}(x_i, x_j) = \sum_k \frac{(p_{ik}^{(t)} - p_{jk}^{(t)})^2}{\varphi^{(0)}(x_k)} \quad (5-35)$$

其中, $\varphi^{(0)}(x_i)$ 可使得高密度的图部分获得更大的权重, 其定义为 $\varphi^{(0)}(x_i) =$

$$\frac{m_i}{\sum_j m_j}, \text{ 节}$$

$$\begin{aligned} \varphi(Y) &= \sum_{i,j} (y_i - y_j)^2 \omega_{ij} = 2Y^T L Y = \sum_{i,j} (y_i^2 + y_j^2 - 2y_i y_j) \omega_{ij} \\ &= \sum_i y_i^2 m_{ii} + \sum_{jj} y_j^2 m_{jj} - 2 \sum_{i,j} y_i y_j \omega_{ij} = 2Y^T M Y - 2Y^T W Y = 2Y^T L Y \end{aligned}$$

点 x_i 的度 m_i 定义为 $m_i = \sum_j p_{ji}$ 。由式(5-35)可以看出具有高的前向转化概率的数据对应较的 Diffusion 距离。这使得 Diffusion 距离对噪声比测地线距离具有更强的鲁棒性, 在低维表示的数据 Y 中, DM 尽可能地保持 Diffusion 距离, 应用随机游走的谱理论, 由特征问题的 d 个非平凡主特征向量, 得到低维

表示 Y :

$$p^{(i)}Y = \lambda Y \quad (5-36)$$

因为图是全连接的,最大的特征值是平凡的,所以其对应的特征向量要去除。这样,低维表示 Y 就是由随后 d 个主特征向量标准化得到。因此,数据的低维表示如下:

$$Y = \{\lambda_2 \nu_2, \lambda_3 \nu_3, \dots, \lambda_{d+1} \nu_{d+1}\} \quad (5-37)$$

5. 随机邻域嵌入(SNE)

SNE 算法的思想也是使得原数据近邻的点降维后还邻近,远离的数据点还彼此远离。首先,定义了一种新的数据近邻关系概率测度公式:

$$p_{ij} = \frac{\exp(-d_{ij}^2)}{\sum_{k \neq i} \exp(-d_{ik}^2)} \quad (5-38)$$

其中, $d_{ij}^2 = \frac{x_i - x_j^2}{2\sigma_i^2}$, σ_i 由经验赋值。

式(5-38)是高维空间邻域数据点间关系的测度公式,同样还可定义低维空间邻域数据点间关系的测度公式为

$$q_{ij} = \frac{\exp(-y_i - y_j^2)}{\sum_{k \neq i} \exp(-y_i - y_k^2)} \quad (5-39)$$

由公式(5-38)、(5-39)利用 Kull-Leibler 散度和来构造匹配损失函数:

$$C = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} = \sum_i KL(p_i | Q_i) \quad (5-40)$$

最小化公式(5-40)得到低维嵌入坐标表示:

$$Y = \underset{Y}{\operatorname{argmin}} C$$

通过梯度下降法迭代求解公式(5-40)的优化问题时,与选择初始点有很大关系,易陷入局部极小,这也是该方法的一个缺点。

6. 最大方差展开(MVU)

MVU 又称为半正定规划(SDE),其思想是保持局部近邻数据点间的距离不变,最大化非近邻数据点的低维表示数据点间的距离,该算法非常巧妙地把局部近邻数据点的距离变为附加约束条件,把最大化非近邻数据间的距离转化为求解半正定规划问题,可用下面的公式表示:

$$\max \sum_{i,j} y_i - y_j^2$$

$$s. t. \sum_i y_i = 0, y_i - y_j^2 = D_{ij} \quad (5-41)$$

其中, $\eta_{ij}=1$, 表示原数据点 x_i 数据点 y_j 为近邻关系, 若为 0 则表示两者不是近邻关系。

式(5-41)的优化问题为非凸的优化问题。为了能便于求解, 做如下变化, 令

$$k_{ij} = y_i \cdot y_j \quad (5-42)$$

则式(5-41)的优化变为半正定优化问题, 即

$$\max \text{Tra}(K)$$

$$s. t. (1) K \geq 0;$$

$$(2) \sum_{i,j} K_{ij} = 0$$

$$(3) K_{ii} - 2K_{ij} + K_{jj} = D_{ij} \eta_{ij} = 1 \quad (5-42)$$

其中, 条件(1) $K \geq 0$ 是要求矩阵 K 为半正定矩阵。

最大方差展开算法步骤总结如下。

(1) 计算每个数据点的 K 近邻, 构造近邻连接图。

(2) 应用半正定规划求解内积矩阵, 并要满足约束条件。

(3) 谱分解内积矩阵, 得到低维坐标表示。

MVU 算法巧妙地把维数简问题与模式识别中应用很广的核方法和半正定规划方法联系起来, 但是算法中半正定规划的计算复杂度较高, 要求数据集的个数一般不超过 2500, 这也极大地限制了该方法的应用。首先, 在 n 个数据点集中随机选择 m 个数据点为标记点(Landmarks)(一般 $m \leq n$); 其次, 应用半正定规划求解标记点的内积矩阵; 最后, 通过式(5-43)计算全部数据的内积矩阵:

$$K \approx QLQ^T \quad (5-42)$$

其中, L 是 $m \times m$ 的标记点内积矩阵, Q 是有求解稀疏线性方程组得到的 $n \times m$ 线性转化矩阵。

7. 等角特征映射(CE)

CE 算法是局部线性嵌入和 Laplace 特征映射的改进方法。为什么这么说呢? 第一, 它们三者保持的性质有些相近, 局部线性算法保持局部重构权重,

Laplace 特征映射算法保持局部数据点的相似度,而 CE 的约束有一些放松,保持局部近邻点间的角度不变,近邻点间的距离可以适当地变化;第二,CE 算法先要应用局部线性算法或 Laplace 特征映射算法把原始高维数据 X 降到 m 维数据 Z ,然后再把 m 维数据 Z 应用等角变化降为想要维数的数据表示。

假设数据 x_i 的两个近邻点为 x_j 和 x_k ,它们三者的低维坐标表示为 y_i, y_j, y_k 。要使数据点 $\{x_i, x_j, x_k\}$ 组成的三角形与相应的低维数据点 $\{y_i, y_j, y_k\}$ 组成的三角形对应角相等,也就是使式(5-43)成立,即

$$\frac{y_i - y_j}{x_i - x_j} = \frac{y_i - y_k}{x_i - x_k} = \frac{y_j - y_k}{x_k - x_j} \quad (5-43)$$

由式(5-43)可知,保持角度相等可等价于各对应边等尺度伸缩。下面给出了度量局部高维空间三角形与低维空间三角形的不相似公式:

$$D_i(s_i) = \sum_{j,k} \eta_{ij} \eta_{ik} (y_j - y_k^2 - s_i x_j - x_k^2)^2 \quad (5-44)$$

其中, $\eta_{ij} = 1$ 表述数据点 x_i 与 x_j 为近邻点,否则为 0, s_i 为伸缩因子。

那么,全局的不相似函数为

$$D(s) = \sum_i D_i(s_i) \quad (5-45)$$

从 m 维数据 Z 到低维表示的线性变化为

$$y_i = L z_i (i=1, 2, \dots, n) \quad (5-46)$$

其中, $L \in \mathbf{R}^{m \times m}$ 是一个广义线性变化矩阵。

把最小化式(5-45)的优化问题转化为半正定规范化问题:

mint

s. t. $P \geq 0$

trace(P) ≥ 1

$$\begin{bmatrix} I & SP \\ (SP)^T & t \end{bmatrix} \geq 0 \quad (5-47)$$

其中, $P = L^T L$, $P = \text{vec}(p)$, 即把矩阵 P 转化为向量 P , I 和 S 为 $m^2 \times m^2$ 的矩阵。

通过式(5-47)的半正定规范化问题求解得到矩阵 P ,从而可得到线性变化矩阵 $L = P^{1/2}$ 。最后,低维空间数据表示式(5-46)计算得到。

5.8 李群机器学习^①

李群机器学习作为机器学习领域的一种新的学习方法,一方面继承流形学习的优点,另一方面借用李群的思想,形成了具有创新特色的学习范式。自2004年提出至今,已引起加拿大、爱尔兰、芬兰、意大利、美国等国内外同行的广泛关注。李群结构是目前学术界公认的对学习问题研究很有用的一套理论工具。从数据分析的角度来说,用机器学习进行数据分析(数据挖掘),其目的就是揭示这些数据具有的规律,从而帮助用户提供解释的依据。李群一方面具有好的数学结构,另一方面物理学家广泛使用李群方法来处理物理学中复杂数据的启发。因此,引进李群理论对机器学习是一种可以探索的新思路。

5.8.1 李群机器学习的基本概念

定义 1

设 G 是一个非空集合,满足如下条件:

(1) G 是一个群;

(2) G 也是一个微分流形;

(3) 群的运算是可微的,即由 $G \times G$ 到 G 的映射 $(g_1, g_2) \rightarrow g_1 g_2^{-1}$ 是可微的映射。则称 G 是一个李群(Liegroup)。

从李群的定义可以看出:李群既是一个群,同时也是一个微分流形。我们知道,流形是点、线、面以及各种高维连续空间概念的推广,而我们在用机器学习方法分析数据时,所有观测数据都是可以和点、线、面等结构建立起对应关系的。从流形的角度看,李群是一种特殊流形,已被物理学家、化学家广泛使用,这充分说明在大量的物理、化学数据中蕴含李群规律,因此,用李群方法去分析这些数据的规律已成为一种必然。所以,研究者充分应用计算机科学与人工智能技术方法从机器学习角度出发,引进李群理论,给出了一种机器学习新方法,即李群机器学习的基本概念,为处理这些复杂的数据提供了新途径。

定义 2 (李群机器学习)

一般用 G 表示输入空间, M 表示输出空间。

^①李凡长,何书萍,钱旭培. 李群机器学习研究综述[J]. 计算机学报,2010,33(7):1115-1126.

令 $G \subseteq R^D, M \subseteq R^d, D > d$ 借用李群的定义将 G 对 M 的左作用可用如下映射 φ 表示:

$$\varphi: G \times M \rightarrow M$$

满足:

(1) 对于所有的 $x \in M, \varphi(e, x) = x$;

(2) 对于所有的 $g, h \in G$ 和 $x \in M, \varphi(g, \varphi(h, x)) = \varphi(gh, x)$ 。右作用是满足条件 $\psi(e, x) = x$ 和 $\psi(\psi(x, g), h) = \psi(x, gh); \psi: G \times M \rightarrow M$ 。

定义 3(轨道的定义)

若 φ 表示 G 在 M 上作用并且 $x \in M$, 则 X 的轨道定义如下:

$$Orb(x) = \{\varphi_g(x) \mid g \in G\} \subset M$$

在有限维情形, $Orb(x)$ 是 M 的浸入子流形。对于 $x \in M, \varphi$ 在 x 处的稳定(或对称)群由 $G_x \stackrel{\text{def}}{=} \{g \in G \mid \varphi_g(x) = x\} \subset G$ 给出, 并且 G 是一个李子群。 $Orb(x)$ 的流形结构由要求双射 $[g] \in G/G_x \rightarrow g \cdot x \in Orb(x)$ 为微分同胚而定义, 由此可以判定 G/G_x 是一个光滑流形。

定义 4(作用的可迁性、有效性和自由性的定义)

(1) 作用具有可迁性, 如果存在唯一的一个轨道, 或等价地, 对于每一个 $x, y \in M$, 有一个 $g \in G$, 使得 $g \cdot x = y$ 。

(2) 作用具有有效性(或者一一对应), 如果 $\varphi_R = id_M$, 蕴含着 $g = e$, 即 $g \rightarrow \varphi_R$ 是一对一的。

(3) 作用具有自由性, 如果它没有不动点, 即 $\varphi_R(x) = x$ 蕴含着 $g = e$, 或者等价的, 如果对于每一个 $x \in M, \varphi_g = \{e\}$, 并且每一个自由的作用是一对一的。

例 1 余伴随作用。 G 在其李代数 $\mathfrak{g}$ 的对偶 $\mathfrak{g}^*$ 上的余伴随作用定义如下: 设 $Ad_g^*: \mathfrak{g}^* \rightarrow \mathfrak{g}^*$ 是 Ad_g 的对偶, 定义为 $\langle Ad_g^* \alpha, \epsilon \rangle = \langle \alpha, Ad_g \epsilon \rangle$, 这里 $\alpha \in \mathfrak{g}^*, \epsilon \in \mathfrak{g}$, 则由 $(g, \alpha) \rightarrow Ad_{g^{-1}}^* \alpha$ 给出的映射:

$\varphi^*: G \times \mathfrak{g}^* \rightarrow \mathfrak{g}^*$ 是 G 在 $\mathfrak{g}^*$ 上的余伴随作用。相应的 G 在 $\mathfrak{g}^*$ 上的余伴随表示记为 $Ad^*: G \rightarrow GL(\mathfrak{g}^*, \mathfrak{g}^*), Ad_{g^{-1}}^* = (T_e(R_g \circ L_{g^{-1}}))^*$ 。

定义 5(轨道空间的定义)

等价类的集合 M/G 即称为轨道空间。

定义 6(学习表达式之间的等价性判定)

设 M 和 N 表示两类学习表达式形成的流形, G 是李群, 并由 $\varphi_R: M \rightarrow M$ 作用在 M 上, 由 $\varphi_R: N \rightarrow N$ 作用在 N 上, 构造一个映射 $f: M \rightarrow N$ 关于这些作用是等价的, 如果对于所有的 $g \in G$, 有 $f \circ \varphi_R = \varphi_R \circ f$, 如果 M/G 和 N/G 都有正规投影光滑浸没的光滑流形, 则等变映射 $f: M \rightarrow N$ 可以诱导出一个光滑映射 $f_G: M/N \rightarrow N/G$ 。

5.8.2 李群机器学习的泛化能力假设公理

李群机器学习泛化能力假设公理, 该公理包括如下几条。

(1) 利用李群、李代数和李子群、李子代数的关系来描述泛化能力, 可定义为: 设 G 是一个李群, 对于 G 的李代数 $\mathfrak{g}$ 的任何一个李子代数 $\mathfrak{h}$, 存在着唯一的一个 G 的连通李子群 H , 使得图 5-14 是可交换的。

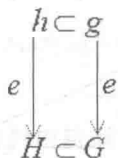


图 5-14 李群、李代数、李子群、李子代数之间的泛化关系图

(2) 利用学习空间中的内积不变性来描述泛化能力, 可定义为: 设 V 是 n 维线性空间, G 是 $GL(V)$ 的李子群, 如果有 $(g(x), g(y)) = (x, y), \forall x, y \in V, g \in G$, 则 V 中内积 (x, y) 称为 G 下不变的。

(3) 利用中心函数来描述泛化能力, 可定义为: 设在紧致李群 G 中取定总体积为 1 的左右不变的黎曼结构, 为其上任何一个中心函数, 则 Weyl 积分公式:

$$\int_G f(g) dg = \frac{1}{|\omega|} \int_{\omega} |f(t)| |Q(t)|^2 dt \quad (5-48)$$

其中, $|\omega|$ 表示 ω 中元素的个数, 当 $t = \exp H (H \in \mathfrak{h})$ 时, $Q(t)$ 可以表示为

$$Q(e^H) = \sum_{\sigma \in \omega} \text{sign}(\sigma) e^{2x_{\sigma}(\sigma(\delta), H)} \quad (5-49)$$

其中, δ 是对于范围 C^0 的正根之和的一半, 即

$$\delta = \frac{1}{2} \sum_{\alpha \in \Delta^+(G)} \alpha' \quad (5-50)$$

5.8.3 李群机器学习的学习模型

李群机器学习系统中有两种学习模型,即代数模型和几何模型,这也就是李群机器学习方法和现有机器学习方法的主要区别之一,如流形学习、统计学习、支持向量机学习、关系学习、决策树学习、贝叶斯学习等,其模型分为代数模型和几何模型。

1. 李群机器学习的代数模型

在李群机器学习系统中,将输入数据 R 做变换,通过 $\theta: R \rightarrow G$ 变换成单参数子群 G ,利用 G 中的右平移变换映射 $R_\theta: R \times G \rightarrow G; (t, g) \rightarrow g \cdot \theta(t)$,它是流形 G 上的一个可微变换群。同样可得左平移变换。由此有{单参数子群}和{左不变流形}是等价的。在通过左不变流形和左不变向量场之间的等价关系,即 $g \cong \{\text{左不变向量场}\} \cong \{\text{左不变流形}\} \cong \{\text{单参数子群}\}$,即可将李群机器学习的代数模型表示为图 5-15 所示的形式:

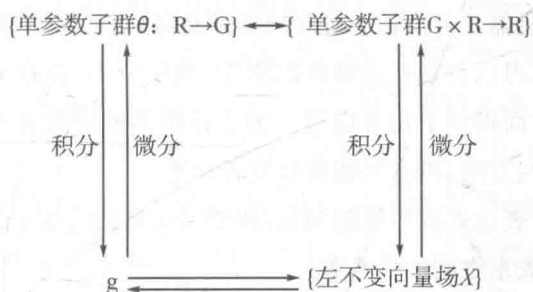


图 5-15 李群机器学习的代数模型

2. 李群机器学习的几何模型

主要根据李群的一些几何性质,如平移不变性、旋转不变性、测地线性性质,给出李群机器学习的几何模型。

一般来说,在一个学习系统中给定一个观测集,将观测集映射成一个精致联通的集合,这样就可以在观测集中的单位点上取一个同构 $Ad(G)$ 作用下不变的内积 $\langle, \rangle$,对于这样取定的内积,再取一组标准正交基,然后用左平移把他们分别扩张成观测集的左不变向量场,这样就可以唯一地在观测集上的每一点的切空间去定内积,使得该内积签好为该点的切空间的标准正交基,这样就构成了一个黎曼空间。

由正交向量场组的左不变性可以看出,所有左平移都是这个李曼空间的等

距变换;由内积的不变性可知,所有的右平移也是他的等距变换。令 $T_a G$ 为单位的切空间,即 G 的李代数 $\mathfrak{g}$, $T_a G$ 为 G 在任意样本点 a 的切空间, dl_a, dr_a 分别是左平移 l_a 和右平移 r_a 在切空间之间诱导的线性映射,由此可将理论机器学习的几何模型表示为图 5-16 所示的形式。

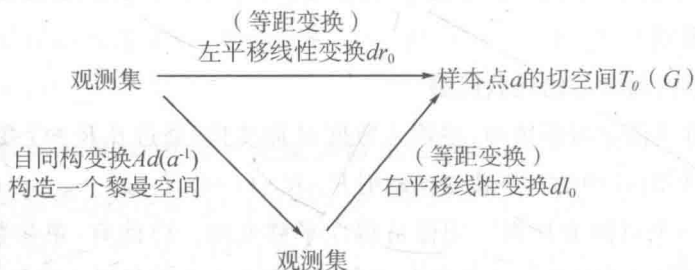


图 5-16 群机器学习的几何模型

5.8.4 李群机器学习小节

通过对李群机器学习假设公理和模型的总结,表明了李群机器学习的基本理论,李群机器学习已分别在计算机图形学、数据分类、药物分子设计、晶体分类、人脸识别等方面得到了初步应用。为了让更多的研究者参与到该领域来,下面列出一些研究方向,供有兴趣的研究者参考。

(1) 李群学习表达式映射机制问题,如学习表达式的表示、学习表达式之间的关系及表达式关系之间的关系等。

(2) 李群深层结构学习机制问题:如泊松结构学习、辛结构学习等。

(3) 更多的现代数学方法应用在机器学习领域,如群范畴方法、层范畴方法等。

(4) 物理学方法在机器学习中的应用等。

第6章 大数据时代的数据挖掘与机器学习

6.1 概论^①

人工授精的过程是从妇女的卵巢中收集卵子,在与丈夫或捐精人的精液结合后产生胚胎,然后从中选择几个移植到妇女的子宫里。关键是要选出那些存活可能性最大的胚胎。选择根据 60 个左右胚胎特征的记录做出,这些特征包括它们的形态、卵母细胞、滤泡和精液样品。特征属性的数量非常大,胚胎学家很难同时对所有属性进行评估,并结合历史数据得出最终结论:这个胚胎是否能够产生一个活的婴儿。在英格兰的一个研究项目中,研究者探索运用机器学习技术,使用胚胎训练数据的历史记录和结果,来做出这个选择。

每年,新西兰奶牛场主都要面临艰难的商业决策:哪些牛应该留在牧场,哪些需要卖到屠宰场。随着饲料储备的减少,每年牧场在接近挤奶季节末期时只留下 1/5 的奶牛。每头牛的生育和牛奶产量的历史数据都会影响这个决定。除此以外还要考虑的因素有:年龄(每头牛都将在八岁后接近生育期的终结)、健康问题、难产的历史数据、不良的性情特征(如咆哮子、跳栅栏)、在下一个季节里不产牛犊。在过去的几年,几百万头牛中的每一头牛都用 700 多个属性记录下来。机器学习正是用来考察成功的农场主在做决定的时候需要考虑哪些因素,不是为了使决策自动化,而是向其他人推广这些农场主的技术和经验。

机器学习是从数据中挖掘知识。它是一个正在萌芽的新技术,范围涉及生与死、从欧洲到两极、家庭和事业,正逐渐引起人们的重视。

我们正在被数据所淹没。存在于这个世界和我们生活中的数据总量似乎在不断地增长,而且没有停止的迹象。个人计算机的普及将那些以前会丢弃的数据保存起来。便宜的、大容量的硬盘推迟了用这些数据做什么的决定,因为我们可以以买更多的硬盘来保存数据。无处不在的电子数据记录了我们的决策,如超市里的商品选择;个人的理财习惯;以及收入和消费。我们以自己的方

^①IAN H. WITTEN E. F. 数据挖掘实用机器学习技术(第一部分:机器学习工具与技术)[M]. 第2版. 北京:机械工业出版社,2006.

式生活在这个世界上,而每一个行为又成为一条数据库里的记录。如今互联网用信息将我们淹没,我们在网上所做的每一个选择都被记录。所有的这些信息记录了个人的选择,而在商业和企业领域存在着数不清的相似案例。我们都知道我们对数据的掌握了解永远无法赶上数据升级的速度。而且在数据量增加的同时,无情地伴随着人们对它的“理解度”的降低。隐藏在这些数据后的是信息,具有潜在用处的信息。而这些信息却很少被显现出来或者被开发利用。

人类从一开始就试图在数据中寻找模式。猎人在动物迁徙的行为中寻找模式;农夫在庄稼的生长中寻找模式;政客在选民的意见上寻找模式;恋人在对方的反应中寻找模式。科学家的工作(像一个婴儿)是理解数据,从数据中找出模式,并用它们来指导在真实世界中如何运作,然后把它们概括成理论,这些理论能够预测出在新的情况下会发什么。企业家的工作是要辨别出机会,就是那些可以转变成有利可图的生意的行为中的一些模式,并且利用这些机会。

在数据挖掘中,电子计算机以电子化的形式存储数据,并且能自动地查询数据,或至少扩增数据。这仍算不得新鲜事。经济学家、统计学家、预测家和信息工程师长久以来相信,存在于数据中的模式能够被自动地找到、识别、确认并能用于预测。该理论的最新发展使得由数据中找出模式的机遇剧增。在最近几年,数据库急剧膨胀,如每天记录顾客选择商品行为的数据库,正把数据挖掘带到新的商业应用技术的前沿。据估计,存储在全出界数据库里的数据量正以每二十个月翻一倍的速度增长。尽管很难从量的意义上真正验证这个数字,但是我们可以从质上把握这个增长速度。随着数据量的膨胀,以及利用机器承担数据搜索工作已变得普通,数据挖掘的机会正在增长。世界正越来越丰富多彩,从中产生的数据淹没了我们,数据挖掘技术成为我们洞察构成数据模式的唯一希望。被充分研究过的数据是宝贵的资源。它能够引导人们去获得新的洞察力,用商业语言讲是获得竞争优势。

数据挖掘就是通过分析存在于数据库里的数据来解决问题。例如,在激烈竞争的市场上,客户忠诚度摇摆问题就是一个经常提到的事例。一个有关客户商品选择以及客户个人资料的数据库是解决这个问题的关键。以前客户的行为模式能够被用来分析并识别那些喜欢选购不同商品和那些喜欢选择同种商品客户的特性。一旦这些特性被发现,它们将被用于当前实际的客户群中,鉴

别出那些善变的客户群体,并加以特殊对待,须知对整个客户群都加以特殊对待的成本是高昂的。同样技术还能够被用来辨别出那些对企业当前提供的服务并不满意,但是有可能对其他的服务感兴趣的客户群,并向他们提供特殊建议,从而推广这些服务。在当代竞争激烈、以客户和服务为中心的市场经济中,如果数据能够被挖掘,它将成为推动企业发展的原材料。

数据挖掘被定义为找出数据中的模式的过程。这个过程必须是自动的或(通常)半自动的。数据的总量是相当可观的,但从中发现的模式必须是有意义的,并能产生出一些效益,通常是经济上的效益。

如何表示数据模式?有价值的模式能够让我们在数据上做出非凡的预测。表示一个模式有两种极端方法:一种是内部结构很难被理解的黑匣子;一种是展示模式结构的透明的匣子,它的结构揭示了模式的结构。我们假设两种方法都能做出好的预测,它们的区别在于被挖掘出的模式能否以结构的形式表现,这个结构是否能够经得起分析,理由是否充分,能否用来形成未来的决策。如果模式能够以显而易见的方法获得决策结构,我们就称他们为结构模式,换句话说,它们能帮助解释有关数据的一些现象。

6.2 机器学习与数据挖掘

“机器学习”是人工智能的核心研究领域之一,其最初的研究动机是为了让计算机系统具有人的学习能力以便实现人工智能,因为众所周知,没有学习能力的系统很难被认为是具有智能的。目前被广泛采用的机器学习的定义是“利用经验来改善计算机系统自身的性能”。事实上,由于“经验”在计算机系统中主要是以数据的形式存在的,因此机器学习需要设法对数据进行分析,这就使得它逐渐成为智能数据分析技术的创新源之一,并且为此而受到越来越多的关注。

“数据挖掘”和“知识发现”通常被相提并论,并在许多场合被认为是可以相互替代的术语。对数据挖掘有多种文字不同但含义接近的定义,例如“识别出巨量数据中有效的、新颖的、潜在有用的、最终可理解的模式的非平凡过程”。其实顾名思义,数据挖掘就是试图从海量数据中找出有用的知识。

大体上看,数据挖掘可以视为机器学习和数据库的交叉,它主要利用机器

学习界提供的技术来分析海量数据,利用数据库界提供的技术来管理海量数据。

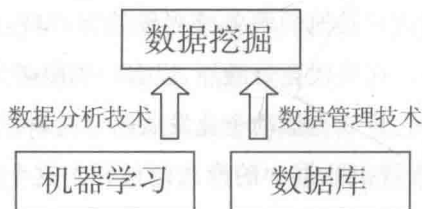


图 6-1 机器学习与数据挖掘的关系示意图

6.2.1 无处不在的机器学习与数据挖掘

随着计算机技术的飞速发展,人类收集数据、存储数据的能力得到了极大的提高,无论是科学研究还是社会生活的各个领域中都积累了大量的数据,对这些数据进行分析以发掘数据中蕴含的有用信息,成为几乎所有领域的共同需求。正是在这样的大趋势下,机器学习和数据挖掘技术的作用日渐重要,受到了广泛的关注。

例如,网络安全是计算机界的一个热门研究领域,特别是在入侵检测方面,不仅有很多理论成果,还出现了不少实用系统。那么,人们如何进行入侵检测呢?首先,人们可以通过检查服务器日志等手段来收集大量的网络访问数据,这些数据中不仅包含正常访问模式还包含入侵模式。然后,人们就可以利用这些数据建立一个可以很好地把正常访问模式和入侵模式分开的模型。这样,在今后接收到一个新的访问模式时,就可以利用这个模型来判断这个模式是正常模式还是入侵模式,甚至判断出具体是何种类型的入侵。显然,这里的关键问题是如何利用以往的网络访问数据来建立可以对今后的访问模式进行分类的模型,而这正是机器学习和数据挖掘技术的强项。

实际上,机器学习和数据挖掘技术已经开始在多媒体、计算机图形学、计算机网络乃至操作系统、软件工程等计算机科学的众多领域中发挥作用,特别是在计算机视觉和自然语言处理领域,机器学习和数据挖掘已经成为最流行、最热门的技术,以至于在这些领域的顶级会议上相当多的论文都与机器学习和数据挖掘技术有关。总的来看,引入机器学习和数据挖掘技术在计算机科学的众多分支领域中都是一个重要趋势。

机器学习和数据挖掘技术还是很多交叉学科的重要支撑技术。例如,生物信息学是一个新兴的交叉学科,它试图利用信息科学技术来研究从 DNA 到基因、基因表达、蛋白质、基因电路、细胞、生理表现等一系列环节上的现象和规律。随着人类基因组计划的实施,以及基因药物的美好前景,生物信息学得到了蓬勃发展。实际上,从信息科学技术的角度来看,生物信息学的研究是一个从“数据”到“发现”的过程,这中间包括数据获取、数据管理、数据分析、仿真实验等环节,而“数据分析”这个环节正是机器学习和数据挖掘技术的舞台。

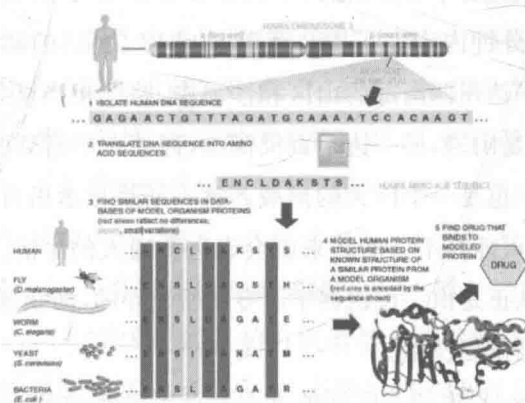


图 6-2 生物信息领域中的数据挖掘示意图

正因为机器学习和数据挖掘技术的进展对计算机科学乃至整个科学技术领域都有重要意义,美国 NASA-JPL 实验室的科学家 2001 年 9 月在 *Science* 上专门撰文指出,机器学习对科学研究的整个过程正起到越来越大的支持作用,并认为该领域将稳定而快速地发展,并将对科学技术的发展发挥更大的促进作用。NASA-JPL 实验室的全名是美国航空航天局喷气推进实验室,位于加州理工学院,是美国尖端技术的一个重要基地,著名的“勇气”号和“机遇”号火星机器人正是在这个实验室完成的。从目前公开的信息来看,机器学习和数据挖掘技术在这两个火星机器人上有大量的应用。

除了在科学研究中发挥重要作用,机器学习和数据挖掘技术和普通人的生活也息息相关。例如,在天气预报、地震预警、环境污染检测等方面,有效地利用机器学习和数据挖掘技术对卫星传递回来的大量数据进行分析,是提高预

报、预警、检测准确性的重要途径;在商业营销中,对利用条形码技术获得的销售数据进行分析,不仅可以帮助商家优化进货、库存,还可以对用户行为进行分析以设计有针对性的营销策略;下面再举两个例子。

公路交通事故是人类面临的重大杀手之一,全世界每年有上百万人丧生车轮,仅我国每年就有约 10 万人死于车祸。美国一直在对自动驾驶车辆进行研究,因为自动驾驶车辆不仅在军事上有重要意义,还对减少因酒后、疲劳而引起的车祸有重要作用。2004 年 3 月,在美国 DARPA(国防部先进研究计划局)组织的自动驾驶车辆竞赛中,斯坦福大学的参赛车辆在完全无人控制的情况下,成功地在 6 小时 53 分钟内走完了 132 英里(约 212 公里)的路程,获得了冠军。比赛路段是在内华达州西南部的山区和沙漠中,路况相当复杂,有的地方路面只有几米宽,一边是山岩,另一边是百尺深沟,即使有丰富驾驶经验的司机,在这样的路段上行车也是一个巨大的挑战。这一结果显示自动驾驶车辆已经不再是一个梦想,可能在不久的将来就会走进普通人的生活。值得一提的是,斯坦福大学参赛队正是由一位机器学习专家所领导的,而获胜车辆也大量使用了机器学习和数据挖掘技术。

Google、Yahoo、百度等互联网搜索引擎已经开始改变了很多人的生活方式,例如很多人已经习惯于在出行前通过网络搜索来了解旅游景点的背景知识、寻找合适的旅馆、饭店等。美国新闻周刊曾经对 Google 有个“一句话评论”:“它使得任何人离任何问题的答案之间的距离只有点击一下鼠标这么远”。现在很少有人不知道互联网搜索引擎的用处,但可能很多人并不了解,机器学习和数据挖掘技术正在支撑着这些搜索引擎。其实,互联网搜索引擎是通过分析互联网上的数据来找到用户所需要的信息,而这正是一个机器学习和数据挖掘任务。事实上,无论是 Google、Yahoo,还是微软,其互联网搜索研究核心团队中都有相当大比例的人是机器学习和数据挖掘专家,而互联网搜索技术目前也正是机器学习和数据挖掘的热门研究话题之一。

6.2.2 机器学习与数据挖掘的发展历程

机器学习是人工智能研究发展到一定阶段的必然产物。从 20 世纪 50 年代到 70 年代初,人工智能研究处于“推理期”,人们认为只要给机器赋予逻辑推

理能力,机器就能具有智能。这一阶段的代表性工作主要有 A. Newell 和 H. Simon 的“逻辑理论家”程序以及此后的“通用问题求解”程序等,这些工作在当时取得了令人振奋的成果。例如,“逻辑理论家”程序在 1952 年证明了著名数学家罗素和怀特海的名著《数学原理》中的 38 条定理,在 1963 年证明了全部的 52 条定理,而且定理甚至比罗素和怀特海证明得更巧妙。A. Newell 和 H. Simon 因此获得了 1975 年图灵奖。然而,随着研究向前发展,人们逐渐认识到,仅具有逻辑推理能力是远远实现不了人工智能的。E. A. Feigenbaum 等人认为,要使机器具有智能,就必须设法使机器拥有知识。在他们的倡导下,20 世纪 70 年代中期开始,人工智能进入了“知识期”。在这一时期,大量专家系统问世,在很多领域做出了巨大贡献。E. A. Feigenbaum 作为“知识工程”之父在 1994 年获得了图灵奖。但是,专家系统面临“知识工程瓶颈”,简单地说,就是由人来把知识总结出来再教给计算机是相当困难的。于是,一些学者想到,如果机器自己能够学习知识该多好!

实际上,图灵在 1950 年提出图灵测试的文章中,就已经提到了机器学习的可能,而 20 世纪 50 年代其实已经开始有机器学习相关的研究工作,主要集中在基于神经网络的连接主义学习方面,代表性工作主要有 F. Rosenblatt 的感知机、B. Widrow 的 Adaline 等。在 20 世纪六七十年代,多种学习技术得到了初步发展,例如以决策理论为基础的统计学习技术以及强化学习技术等,代表性工作主要有 A. L. Samuel 的跳棋程序以及 N. J. Nilson 的“学习机器”等,20 多年后红极一时的统计学习理论的一些重要结果也是在这个时期取得的。在这一时期,基于逻辑或图结构表示的符号学习技术也开始出现,代表性工作有 P. Winston 的“结构学习系统”、R. S. Michalski 等人的“基于逻辑的归纳学习系统”、E. B. Hunt 等人的“概念学习系统”等。

1980 年夏天,在美国卡内基梅隆大学举行了第一届机器学习研讨会;同年,《策略分析与信息系统》连出三期机器学习专辑;1983 年,Tioga 出版社出版了 R. S. Michalski、J. G. Carbonell 和 T. M. Mitchell 主编的《机器学习:一种人工智能途径》,书中汇集了 20 位学者撰写的 16 篇文章,对当时的机器学习研究工作进行了总结,产生了很大反响;1986 年,*Machine Learning* 创刊;1989 年,*Artificial Intelligence* 出版了机器学习专辑,刊发了一些当时比较活跃的研究

究工作,其内容后来出现在 J. G. Carbonell 主编、MIT 出版社 1990 年出版的《机器学习:风范与方法》一书中。总的来看,20 世纪 80 年代是机器学习成为一个独立的学科领域并开始快速发展、各种机器学习技术百花齐放的时期。

R. S. Michalski 等人把机器学习研究划分成“从例子中学习”“在问题求解和规划中学习”“通过观察和发现学习”“从指令中学习”等范畴;而 E. A. Feigenbaum 在著名的《人工智能手册》中,则把机器学习技术划分为四大类,即“机械学习”“示教学习”“类比学习”“归纳学习”。机械学习也称为“死记硬背式学习”,就是把外界输入的信息全部记下来,在需要的时候原封不动地取出来使用,这实际上没有进行真正的学习;示教学习和类比学习实际上类似于 R. S. Michalski 等人所说的“从指令中学习”和“通过观察和发现学习”;归纳学习类似于“从例子中学习”,即从训练例中归纳出学习结果。20 世纪 80 年代以来,被研究得最多、应用最广的是“从例子中学习”(也就是广义的归纳学习),它涵盖了监督学习(例如分类、回归)、非监督学习(例如聚类)等众多内容。下面我们对这方面主流技术的演进做一个简单的回顾。

在 20 世纪 90 年代中期之前,“从例子中学习”的一大主流技术是归纳逻辑程序设计(Inductive Logic Programming),这实际上是机器学习和逻辑程序设计的交叉。它使用 1 阶逻辑来进行知识表示,通过修改和扩充逻辑表达式(例如 Prolog 表达式)来完成对数据的归纳。这一技术占据主流地位与整个人工智能领域的发展历程是分不开的。如前所述,人工智能在 20 世纪 50 年代到 80 年代经历了“推理期”和“知识期”,在“推理期”中人们基于逻辑知识表示、通过演绎技术获得了很多成果,而在知识期中人们基于逻辑知识表示、通过领域知识获取来实现专家系统,因此,逻辑知识表示很自然地受到青睐,而归纳逻辑程序设计技术也自然成为机器学习的一大主流。归纳逻辑程序设计技术的一大优点是它具有很强的知识表示能力,可以较容易地表示出复杂数据和复杂的数据关系。尤为重要的是,领域知识通常可以方便地写成逻辑表达式,因此,归纳逻辑程序设计技术不仅可以方便地利用领域知识指导学习,还可以通过学习对领域知识进行精化和增强,甚至可以从数据中学习出领域知识。事实上,机器学习在 20 世纪 80 年代正是被视为“解决知识工程瓶颈问题的关键”而走到人工智能主舞台的聚光灯下的,归纳逻辑程序设计的一些良好特性对此无疑居

功至伟。S. H. Muggleton 主编的书对 90 年代中期之前归纳逻辑程序设计方面的研究工作做了总结。然而,归纳逻辑程序设计技术也有其局限,最严重的问题是由于其表示能力很强,学习过程所面临的假设空间太大,对规模稍大的问题就很难进行有效的学习,只能解决一些“玩具问题”。因此,在 20 世纪 90 年代中期后,归纳程序设计技术方面的研究相对陷入了低谷。

20 世纪 90 年代中期之前,“从例子中学习”的另一大主流技术是基于神经网络的连接主义学习。连接主义学习技术在 20 世纪 50 年代曾经历了一个大发展时期,但因为早期的很多人工智能研究者对符号表示有特别的偏爱,例如 H. Simon 曾说人工智能就是研究“对智能行为的符号化建模”,因此当时连接主义的研究并没有被纳入主流人工智能的范畴。同时,连接主义学习自身也遇到了极大的问题,M. Minsky 和 S. Papert 在 1969 年指出,(当时的)神经网络只能用于线性分类,对哪怕“异或”这么简单的问题都做不了。于是,连接主义学习在此后近 15 年的时间内陷入了停滞期。直到 1983 年,J. J. Hopfield 利用神经网络求解 TSP 问题获得了成功,才使得连接主义重新受到人们的关注。1986 年,D. E. Rumelhart 和 J. L. McClelland 主编了著名的《并行分布处理——认知微结构的探索》一书,对 PDP 小组的研究工作进行了总结,轰动一时。特别是 D. E. Rumelhart、G. E. Hinton 和 R. J. Williams 重新发明了著名的 BP 算法,产生了非常大的影响。该算法可以说是最成功的神经网络学习算法,在当时迅速成为最流行的算法,并在很多应用中都取得了极大的成功。与归纳逻辑程序设计技术相比,连接主义学习技术基于“属性—值”的表示形式(也就是用一个特征向量来表示一个事物;这实际上是命题逻辑表示形式),学习过程所面临的假设空间远小于归纳逻辑程序设计所面临的空间,而且由于有 BP 这样有效的学习算法,使得它可以解决很多实际问题。事实上,即使在今天,BP 仍然是在实际工程应用中被用得最多、最成功的算法之一。然而,连接主义学习技术也有其局限,一个常被人诟病的问题是“试错性”。简单地说,在此类技术中有大量的经验参数需要设置,例如神经网络的隐层节点数、学习率等,夸张一点说,参数设置上差之毫厘,学习结果可能谬以千里。在实际工程应用中,人们可以通过调试来确定较好的参数设置,但对机器学习研究者来说,对此显然是难以满意的。

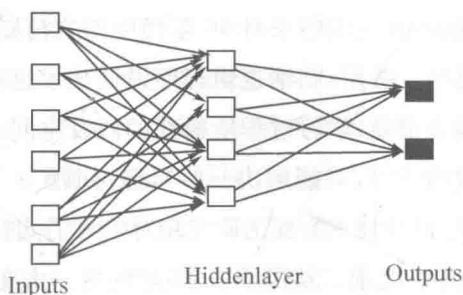


图 6-3 神经网络学习示意图

20 世纪 90 年代中期,统计学习粉墨登场并迅速独占鳌头。其实早在 20 世纪 60、70 年代就已经有统计学习方面的研究工作,统计学习理论在那个时期也已经打下了基础,例如 V. N. Vapnik 早在 1963 年就提出了“支持向量”的概念,他和 A. J. Chervonenkis 在 1968 年提出了 VC 维,在 1974 年提出了结构风险最小化原则等,但直到 90 年代中期统计学习才开始成为机器学习的主流技术。一方面,这是由于有效的支持向量机算法在 90 年代才由 B. E. Boser、I. Guyon 和 V. N. Vapnik 提出,而其优越的性能也是到 90 年代中期才在 T. Joachims 等人对文本分类的研究中显现出来;另一方面,正是在连接主义学习技术的局限性凸显出来之后,人们才把目光转向了统计学习。事实上,统计学习与连接主义学习有着密切的联系,例如 RBF 神经网络其实就是一种很常用的支持向量机。

在支持向量机被普遍接受后,支持向量机中用到的核(kernel)技巧被人们用到了机器学习的几乎每一个角落中,“核方法”也逐渐成为机器学习的一种基本技巧。但其实这并不是一种新技术,例如 Mercer 定理是在 1909 年发表的,核技巧也早已被很多人使用过,即使只考虑机器学习领域,至少 T. Poggio 在 1975 年就使用过多项式核。如果仔细审视统计学习理论,就可以发现其中的绝大多数想法在以往机器学习的研究中都出现过,例如结构风险最小化原则实际上就是对以往机器学习研究中经常用到的最小描述长度原则的另一个说法。但是,统计学习理论把这些有用的片段整合在同一个理论框架之下,从而为人们研制出泛化能力有理论保证的算法奠定了基础,与连接主义学习的“试错法”相比,这是一个极大的进步。然而,统计学习也有其局限,例如,虽然理论上

说,通过把原始空间利用核技巧转化到一个新的特征空间,再困难的问题也可以容易地得到解决,但如何选择合适的核映射,却仍然有浓重的经验色彩。另外,统计学习技术与连接主义学习技术一样是基于“属性—值”表示形式,难以有效地表示出复杂数据和复杂的数据关系,不仅难以利用领域知识,而且学习结果还具有“黑箱性”。此外,传统的统计学习技术往往因为要确保统计性质或简化问题而做出一些假设,但很多假设在真实世界其实是难以成立的。如何克服上述缺陷,正是很多学者正在关注的问题。

机器学习之所以备受瞩目,主要是因为它已成为智能数据分析技术的创新源之一。但是机器学习还有一个不可忽视的功能,就是通过建立一些关于学习的计算模型来帮助人们了解“人类如何学习”。例如,P. Kanerva 在 20 世纪 80 年代中期提出 SDM(Sparse Distributed Memory)模型时并没有刻意模仿人脑生理结构,但后来的研究发现,SDM 的工作机制非常接近于人类小脑,这为理解小脑的某些功能提供了帮助。自然科学研究的驱动力归结起来无非是人类对宇宙本源、物质本性、生命本质、自我本识的好奇,而“人类如何学习”无疑是一个有关自我本识的重大问题。从这个意义上说,机器学习不仅在信息科学中占有重要地位,还有一定的自然科学色彩。与此不同,数据挖掘则是一个直接为实际应用而生的学科领域。20 世纪 60 年代,早期的数据库问世,人们开始利用计算机对数据进行管理;到了 70 年代之后,随着关系数据库的出现和发展,人们管理数据的能力越来越强,收集存储的数据也越来越多。如果只利用数据库进行一些简单的事务处理,显然没有对数据进行充分的利用,从数据中挖掘出有用的知识,才可以更好地实现数据的价值。

1989 年 8 月,第 11 届国际人工智能联合会议(IJCAI'89)在美国底特律举行,GTE 实验室的 G. Piatetsky-Shapiro 在 J. G. Carbonell、W. Frawley、K. Parsaye、J. R. Quinlan、M. Siegel、R. Uthurusamy 等人的支持下,组织了一个名为“在数据库中发现知识”的研讨会,这个研讨会后来被认为是数据挖掘成为一个领域的标志。早期人们一直称其为“数据挖掘与知识发现”,但随着该领域的发展壮大,越来越多的人直接称其为数据挖掘。值得注意的是,数据挖掘的对象早就不限于数据库,而可以是存放在任何地方的数据,甚至包括 Internet 上的数据。

数据挖掘受到了很多学科领域的影响,其中数据库、机器学习、统计学无疑影响最大。粗糙地说,数据库提供数据管理技术,机器学习和统计学提供数据分析技术。由于统计学界往往醉心于理论的优美而忽视实际的效用,因此,统计学界提供的很多技术通常都要在机器学习界进一步研究,变成有效的机器学习算法之后才能再进入数据挖掘领域。从这个意义上说,统计学主要是通过机器学习来对数据挖掘发挥影响,而机器学习和数据库则是数据挖掘的两大支撑技术。

一方面,从数据分析的角度来看,绝大多数数据挖掘技术都来自机器学习领域。但能否认为数据挖掘只不过就是机器学习的简单应用呢?答案是否定的。一个重要的区别是,传统的机器学习研究并不把海量数据作为处理对象,很多技术是为处理中小规模数据设计的,如果直接把这些技术用于海量数据,效果可能很差,甚至可能用不起来。因此,数据挖掘界必须对这些技术进行专门的、不简单的改造。例如,决策树是一种很好的机器学习技术,不仅有很强的泛化能力,而且学得结果具有一定的可理解性,很适合数据挖掘任务的需求。但传统的决策树算法需要把所有的数据都读到内存中,在面对海量数据时这显然是无法实现的。为了使决策树能够处理海量数据,数据挖掘界做了很多工作,例如通过引入高效的数据结构和数据调度策略等来改造决策树学习过程,而这其实正是在利用数据库界所擅长的数据管理技术。实际上,在传统机器学习算法的研究中,在很多问题上如果能找到多项式时间的算法可能就已经很好了,但在面对海量数据时,可能连 $O(n^3)$ 的算法都是难以接受的,这就给算法的设计带来了巨大的挑战。

另一方面,作为一个独立的学科领域,必然会有一些相对“独特”的东西。对数据挖掘来说,这就是关联分析。简单地说,关联分析就是希望从数据中找出“买尿布的人很可能会买啤酒”这样看起来匪夷所思但可能很有意义的模式。如果在 100 位顾客中有 20 位购买了尿布,购买尿布的 20 位顾客中有 16 位购买了啤酒,那么就可以写成“尿布 $\rightarrow$ 啤酒(支持度=20%,置信度=80%)”这样的一条关联规则。挖掘出这样的规则可以有很多用处,例如商家可以考虑把尿布展柜和啤酒展柜放到一起以促进销售。实际上,在面对少量数据时关联分析并不难,可以直接使用统计学中有关相关性的知识,这也正是机器学习界没有

研究关联分析的一个重要原因。关联分析的困难其实完全是由海量数据造成的,因为数据量的增加会直接造成挖掘效率的下降,当数据量增加到一定程度,问题的难度就会产生质变,例如,在关联分析中必须考虑因数据太大而无法承受多次扫描数据库的开销、可能产生在存储和计算上都无法接受的大量中间结果等,而关联分析技术正是围绕着“提高效率”这条主线发展起来的。在 R. Agrawal 等人首先对关联规则挖掘进行研究之后,大批学者投身到这方面的研究中并产生了很多成果,代表性工作有 R. Agrawal 和 R. Srikant 的 Apriori 算法以及 J. Han 等人的 FP-Growth 算法等。

6.3 数据挖掘工具^①

当面临商业挑战的时候,人们需要有工具来解决相应的问题,并发现创造价值的方式。在分析论坛上,解决这些问题的讨论经常以应当使用何种工具的决策开始。这是工作开始前,除了考虑数据存储于何处和如何进行分析之外必须考虑的第三个因素。本节详述一些普遍应用到的工具,其中一些工具笔者非常熟悉,实际上还对其发展起了一定作用。其他工具笔者就只是作为工具加以使用,就像你或你的团队一样,还有一些工具笔者本人也没有使用过,但是它们也应当被考虑到。笔者力图对每个工具的优缺点进行公允的评估。

6.3.1 Weka

Weka(Waika to Environment for Knowledge Analysis)是一个开源数据挖掘产品,完全在 Java 运行,主要是由新西兰的怀卡多(Waikato)大学开发。Weka 由于其大量极为先进的训练算法、其工作流的图形用户界面(GUI),以及与数据可视化工具的结合而广为人知。Weka 允许用户通过被设计用于进行富有成效的数据分析的 GUI、命令行接口、Java 应用编程界面(API)使用其复杂的数据挖掘程序。但是,Weka 用于大数据分析并不是很好,因为它要受通常在单机上可用的 RAM 资源的限制。64 位操作系统的用户将可以利用到大得多的 RAM 资源,而 Weka 的文档能指导用户进行数据预处理以及算法筛

^①IAN H, WITTEN E F. 数据挖掘实用机器学习技术(第二部分:weka 机器学习平台)[M]. 第2版. 北京:机械工业出版社,2006.

选,以便在分析前进行大数据抽样。由于在 Weka 中已有的最强大的算法中,许多算法在其他没有定制软件开发的环境中是不可用的,抽样可能是 Weka 案例应用的最佳实践。在最近的版本中,Weka 已经在多线程和简单的多任务处理方面取得长足的进展。在 Weka3.6 中,一些分类器可以用多折交叉检验同步训练模型,而 Weka 服务器部署考虑到了在单机或集群上并发运行数据挖掘任务的情况。Weka 程序还为两个基于 Java 的大数据分析环境[Pentaho 和 MOA(Massive Online Analysis,大规模在线分析)]提供高级分析组件。对于可能不具有 Java 编程经验并且需要迅速证明其价值的人而言,Weka 是一个好选择。

6.3.2 Java 和 JVM 语言

对于想要从 Scatch 定制分析平台的组织而言,Java 和许多其他在 Java 虚拟机(Java Virtual Machine,JVM)上运行的语言是普遍的选择。对于大型、并发和联网应用,Java 相比其他低级语言具有较大的开发优势。它没有过度牺牲性能,特别是在大数据分析领域的性能。虽然在本机硬件上直接执行的语言,特别是 FORTRAN 和 C 语言,在受 RAM 和 CPU 限制的计算上仍然超过 Java,但是 Java 平台的技术发展已经将其性能在输入/输出和受网络限制的流程方面提高到几乎与本机语言一致,就像许多核心的开源大数据应用一样。Apache Hadoop 可能是最被认可的基于 Java 的大数据环境,它在 2008 年击败许多其他技术和 2009 大字节排序基准。它是第一个击败官方基准的基于 Java 的技术。

作为一个通用编程语言,Java 的优缺点在许多论坛上被广为讨论,在这里我们对其做简短评述。过去,分析应用程序的开发者经常会避免使用 Java,因为其详尽的运行时库对于数值程序是不必要的,而且 JVM 平台的内存管理会导致不可接受的减速。这里只是举出其中两个因素。由于分析应用程序在规模和复杂度上增长显著,开发时间成为更加严重的问题,而受内存与 CPU 限制的性能也就变得没那么重要了。在开发数据挖掘应用程序的特定环境下,Java 因为其开发效率而具有优势:它有丰富的程序库、许多应用程序框架、对同步和通信网络的内在支持,以及先前存在的作为数据挖掘功能的基础的开源代码

(如 Mahout 和 Weka 库)。虽然在性能上有所损失,但是 JVM 平台具有许多开发优势。除了可移植性,JVM 还提供了内存管理、内存概要和自动化异常处理。

Scala 和 Clojure 是更新型的语言,它们同样在 JVM 上运行并被用于数据挖掘应用程序中。Scala 是一个开源语言,由瑞士的洛桑综合工科学学校开发。它首次在 2003 年发布,其目标是对 Java 的改进。

Scala 常常被用于构建数据挖掘应用,因为它通过与 Java 运行环境的交互运作,既实现了 Java 的开发效率,又增加了函数式语言,而且语法全都更加简洁。Scala 允许开发人员在函数式语言和面向对象(OO, object-oriented)的编程范式之间切换,而 Java 的语法结构更倾向于执行 OO 设计。函数式语言被认为对数据挖掘应用更加有利,因为它处理内存的方式与 OO 和过程式语言不同。例如,过程式语言和 OO 多线程应用的一个主要问题是阻止了多线程同时改变同一变量。函数式编程语言通过不改变变量避免了这一情况。即使应用没有那么普遍,函数式语言仍然是一个很有价值的工具,不应当被忽视。Scala 很受欢迎,已经被用于部署像 Akka 和 Spark 这样的主要开源项目。Akka 是一个在 JVM 上构建大型、并行和分布式应用的工具包,而 Spark 是一个来自加州大学伯克利分校 AMP 实验室的高性能数据挖掘环境。Scala 同样已经商用,被四方网(Foursquare)、卫报(the Guardian)、人际关系网(LinkedIn)、Novell 公司、西门子公司(Siemens)、索尼公司(Sony)、Twitter 及其他公司用于各种用途。

Clojure 由 RichHickey 编写并于 2007 年发布。它是一个函数式语言,也是 Lisp 编程语言的变种,有额外的脚本和并发功能。虽然它并不是专门设计用来进行大数据应用的,但这些本质特征使得 Clojure 成为用于构建数据分析软件的一个非常优秀的备用工具。与程式语言和 OO 语言相比较,Clojure 默认数据结构不可变,从而删除了整个类别的线程安全设计问题。Clojure 处理数据的方式非常全面,它提供给开发人员许多便利的方法来处理广泛的数据架构集合。它从 Lisp 继承了将代码看作数据的概念。特别是与 C 语言和 FORTRAN 相比较,这个数据序列方法排除了一类与边界条件有关的常见编程错误。Clojure 还从 Lisp 中继承了简洁的语法、惰性计算和高效的宏功

能。宏功能考虑到了特定领域语言(DSLs, domain-specific languages)的产生,而集成了SQL(Clojure QL)和Hadoop(Cascalog)的DSLs也已经建立。当Clojure的独特优势结合了JVM平台和Java程序库访问的优势,它就成为一个强大的通用编程资产。Clojure的首个采用者通常是较为小型的公司和新创企业,但这些公司在技术或分析上较为先进。不过据报道,大型企业和机构,例如花旗集团(Citigroup)和普朗克研究所(the Max Planck Institute),也已经应用了Clojure。

6.3.3 R语言

R语言是设计用于统计分析的第四代开源编程语言。R语言由商业性S语言发展而来,而S语言则是贝尔实验室从20世纪70年代晚期开始开发的。R语言是被新西兰奥克兰大学的研究人员从S和SPLUS(另一个现在由TIBCO公司授权的商业工具)中移植出来,并首次于1996年面世。

R语言在学术团体中非常流行,笔者也是在其中首次接触到它。用它来测试没有时间或没有能力自己写代码的新分析并完成布置的家庭作业。那时候,这个软件还比笔者用到的其他软件在制图功能上要优越得多。笔者在研究生院期间以及在美国人口调查局工作的时候都使用了这个制图功能。虽然这些制图功能现在仍然在R语言中存在,但与其他软件相比已经失去了其竞争优势。

R在广泛和日益增长的数据科学界中的位置已经非常突出,在学术工具中被广泛应用,在私营部门也被日益接受。众所周知,大型公司,如美国银行(Bank of America)、谷歌(Google)、洲际酒店(Intercontinental Hotels)、默克公司(Merck)、辉瑞制药(Pfizer)和壳牌公司(Shell)都已经出于各种各样的目的应用了R语言。2013年9月TIOBE对编程语言的综合评级中将R放在全部开发语言中最受欢迎的第18位,级别远远高于S语言和S-SPLUS,与像SAS(第21位)和MATLAB(第19位)这样的商业解决方案非常接近(TIOBE, 2013)。R同样也可以高度定制化。R语言有上千种扩展,到2009年大约有超过1600种。扩展包将从语音分析到基因科学、文本挖掘的每个内容都集成到R语言的基线分析功能中。R语言还有令人印象深刻的图表、免费且优美的集

成开发环境 (IDEs, integrated development enviroments)、对许多通用语言的编程访问、大受欢迎的包括 MATLAB 和 SAS 在内的专有分析解决方案的界面,甚至还有来自革命分析公司 (Revolution Analytics) 的商务支持。

R 语言在受欢迎度和功能上经历了一个迅速的发展,在过去的 R 语言版本中,内存问题使得处理大型数据非常困难,现在这个问题已经得到解决。最近,通过集群计算和 Hadoop 界面规避剩余内存限制的重大开发工作正在进行中。R 语言的更新周期很短,使得其消费者可以通过像 Snow、Multicore 和 Paralled 这样的办公包和像 RHadoop 和 RHIPE 这样最近才开始的项目应用这些工作成果。除了 RHadoop 和 RHIPE,这些办公包并不适合真正的大数据分析,但适用于“高度并行”类别的 CPU 密集型计算和内存密集型计算,这些计算通过构建 R 语言的 lappy() 函数来执行,这个函数被应用于列表中的所有元素。

Snow 应用于在计算机集群中运行 R 计算。它使用 sockets、MPI、PVM 或 Net Work Spaces 在集群节点之间相互沟通。Snow 包利用传统主从并行体系架构,在主节点的内存中保存结果,需要 snow 函数调用的参数加载到内存。这样,由于内存限制,程序包主要用于 CPU 密集型计算,例如模拟、靴襻法 (Bootstrapping) 和某些机器学习算法。

Multicore 是一个容易安装和部署的程序包,它将在一台 POSIX 兼容机器 (OSX, Linux 或 UNIX) 上运行的连续的 R 程序分割成在相同机器上的多线程过程。这些用 Multicore 同步执行的程序受到单机 RAM 和其他共享内存限制的制约。

Parallel 是一个并行执行包,它在 R 2.14 或更高版本中成为标准配置。它将开箱即用的并行执行延伸到 POSIX 兼容操作系统的集群水平和 Windows 的多处理器水平。Parallel 与 Snow 和 Multicore 有许多共同的算法、函数和局限性,可以应用 Snow 集群对象。

Rhadoop 和 Rhipe 允许使用 R 语言对 Hadoop 进行编程访问,amap 和 reduce 任务可以通过 R 在 Hadoop 集群中运行。

6.3.4 Python

从 20 世纪 80 年代晚期 Python 的初始时期起,它就被设计为一个可扩展

的高级语言,具有大规模标准程序库和简明的语法。Python 可以被用于交互或编程,它经常被部署在脚本撰写、数值分析、OO 通用开发和网络应用开发中。Python 是一种解释性语言,通常在执行计算密集型任务时会比本地编译语言和 JVM 语言慢。但是,Python 程序在运行时可以通过许多 API 文件调用本地编译的 FORTRAN、C 和 C++。Python 程序可以是多线程的。就像 JVM 语言,Python 平台会进行内存管理和异常处理,将开发人员从这方面解放出来。Python 语言也已经在一个叫作 Jython 的项目中被移植到 JVM 平台。

Python 的通用编程优势,以及它的许多数据库、数学库和图形库,使其非常适合于数据探索和数据挖掘领域的项目。Python 的内在文件处理和文本操作能力,还有其与大部分 SQL 和 NOSQL 数据库相连接的能力,使得 Python 程序能相当容易地加载和处理原始数据和表格数据。数学模型和统计模型可以用 Scipy 库和 Numpy 库实现,它们能够强有力地支持线性代数、数值分析和统计操作。新出现的 Python 库,如 Orange 和 Pattern,提供高级数据挖掘 API,而 Matplotlib 库能够产生复杂微妙的数据可视化。Python 也可以用 Hadoop 的流工具来完成 MapReduce 任务中的 map 和 reduce 指令。

最近,随着 scikit-learn 工具包的成熟,Python 已经在数据挖掘和数据科学团体中得到更多应用。该工具包当前是 0.14 版本,但提供了许多有用的算法和处理技术。Python 的 sdkit 在用户界面上并没有超越 IDE 编程,它提供给用户灵活性,但会让最初的学习曲线相当陡峭,并降低了常规任务的效率。

6.3.5 SAS

SAS 是市场上领先的分析软件。在 2013 年 IDC 报告中,SAS 不仅具有显著市场份额,而且比其后的 16 名供应商加在一起的市场份额还高。SAS 系统首先在 1976 年销售,其 4 个创建者中的两位仍然活跃在公司中 Jim Goodnight 博士和 John Sail 博士。SAS 系统被分割为许多产品领域,包括统计、运筹学、数据管理、数据访问引擎和商业智能(BI)。对于本书的主题,最相关的产品是 SAS/STAT、SAS Enterprise Miner 和 SAS 文本分析组件。在最近的《弗里斯特(Forrester Wave)》和“魔力象限(Gartner Magic Quadrant)”中,SAS 已经被

评为预测建模和数据挖掘领域的领先供应商。这些评定揭示了该软件能力的广度和深度。在我的经验中,没有哪个分析挑战不能用 SAS 来应对。在为公司工作将近 10 年后,笔者更好地理解了开发团队的细致工作和杰出贡献,以及他们为了确保软件质量、满足客户要求、预计客户未来需求而付出的巨大努力。

SAS 系统被分成两个主要领域:执行分析的程序和允许用户操作数据的第四代语言。这被称为 DATAStep。

SAS 的独到成就之一是在每个版本中都保持着反向兼容性。我在美国人口统计局亲眼看见过这一情形。我们有已经运行良好超过 15 年之久的 SAS 程序,它们给出的结果能够被每一个 SAS 的更新程序所验证。这一点非常重要,因为计算机程序的最初创建者经常会迁移到其他部门或者在某些时候不再服务于统计局。

SAS 应用了一个全面的方案,在为个人客户环境提供灵活性的同时也能提供最佳计算性能。这个通常被称为 PROC 的过程是为特定分析而创建的模块,就像 REG 程序是为回归模型所创建的,或者 UNIVARIATE 程序是为了对单个变量进行描述性统计而出现的那样。让我们看一个简单的例子,即进行一项计算给定变量均值的任务。

这可以在任何编程语言中简单地实现,因为求均值非常简单: $\sum_{i=1}^n i/n$ 。这可以用如下代码的数据步骤进行计算:

```
Dataa;
Setbend=last;
RetainX;
Sum+X;
If last then mean=sum/n;
Run;
```

这个例子展示了如何从数据集 b 中生成一个新的数据集。Retain 语句告诉 SAS 当转移到一个新的观测值(行)的时候保存数值; if 语句告诉 SAS 如果这是数据集中的最后一个观测值,就将合计数除以观测值数量,该观测值数量被存储在自动变量 n 中。这个代码并不难写,也不难解释,但对于 SAS 用户还有一个更好的可选方法:

```
Procmeansdata=b;
```

```
Varx;
```

```
Run;
```

除了更加简短,这个代码也更容易读,它使用特定代码而不是通用代码,而我不需要每增加一个新的列就复制一遍数据。该程序同时也能处理和记录大量在你开发自己的软件时非常重要的行为(例如,处理 tie、缺失值或其他数据质量问题)。SAS 仅仅在 SAS/STAT 产品中就有 82 个程序。有了这近百个程序,就能准确执行具体分析任务。SAS 与其他软件包相比的一个主要优势是文档化。SAS/STAT 有超过 9300 页的文档,Enterprise Miner 则有超过 2000 页文档。

在可能的时候,SAS 系统在内存中处理数据,并用系统页面文件有效管理任何规模的数据。在最近几年,SAS 在其架构上已经有很大变化,从而能更好地利用处理能力,降低现代计算集群中每 FLOP(每秒浮点运算速度)的价格。

6.3.6 数据挖掘工具 Weka 和 R 的比较分析

Weka 和 R 是两个突出的开放源码分析软件系统。这两个都来自学术界,但有不同的目标和重点。R 来自统计界,是一个通用分析统计环境,Weka 的起源是在计算机科学,因此专门为机器学习和数据挖掘而设计。在选择分析软件时,你需要仔细考虑你的数据挖掘的目标范围内的各种因素,包括预测潜在部署模型。Weka 的基础是 100% 的 Java,促进简单集成和部署。Weka 提供了技术,广阔的选择数据挖掘和机器学习。R 是一个通用的统计环境,拥有设施。Weka 无疑是更用户友好,有熟悉点的点击图形用户界面。而 R 本质上是一种函数式编程语言。

R 里有很多机器学习的函数和包,不过 Weka 里提供的函数更全面更集中。所以通常在 R 中准备好训练的数据(如:提取数据特征……);整理成 Weka 需要的格式(*.arff);在 Weka 里做机器学习(如:特征选择、分类……);从 Weka 的预测结果计算需要的统计量(如:sensitivity, specificity, MCC……)。

Weak 和 R 的具体比较见下表:

表 6-1 Date exploration/Visualization

Fature	Weka	R
Descriptive statiatics	✓	✓
Frequency table	✓	✓
Scatter plot	✓	✓
Scatter plot matrices	✓	✓
Histonrams	✓	✓
Tree/Graph visualization	✓	✓
Boxplots		✓
Precision/recall curve	✓	✓
Lift chart	✓	✓
ROC curve	✓	✓

表 6-2 Data Preparation

Fature	Weka	R
Sampling		
Oversampling/balanceing	✓	✓
Random	✓	✓
Stratified	✓	✓
Discretization(binning)		
Equal width	✓	✓
Equal frequency	✓	✓
Supervised	✓	
Recorder fields	✓	✓
Identifier fields	✓	✓
Normalization/standar dization	✓	✓

续表

Fature	Weka	R
Binarization	✓	✓
Derived fields	✓	✓
Outlier detection	✓	✓
Principal components	✓	✓
Random projections	✓	✓
Attribute selsction	✓	✓
Trees		
ID3	✓	
C4.5	✓	
CART	✓	✓
Decision stumps	✓	✓
Random forests	✓	✓
Best first tree	✓	
Logistic model trees	✓	
MS model tree	✓	
Alternation decision trees		
Interactive tree construction		
KNN trees		✓
Rules		
Decision table	✓	
RIPPER	✓	
Conjunctive rule	✓	
M5 Rules	✓	

续表

Fature	Weka	R
PART	✓	
Ripplr down rules(Ridor)	✓	
NNge	✓	
OneR	✓	
Ensmeble Learning		
Adaboost	✓	✓
LogitBoost	✓	✓
Additive regression	✓	✓
Bagging	✓	✓
Stacking	✓	
Grading	✓	
MultiBoost	✓	
Voted classifier	✓	
MetaCoat	✓	
Ensembles of nested dichotomiles	✓	
Clustering		
EM	✓	✓
KMeans	✓	✓
XMeams	✓	
COBWEB(hierarchical)	✓	
OPTICS	✓	
Farthest first clustering	✓	
Hierarchical clustering		✓

续表

Fature	Weka	R
Agglomerative nesting		✓
Fuzzy C-means clustering		✓
Bagged clustering		✓
Cluster ensembles		✓
Convex clustering		✓
Association rules		
Apriori	✓	✓ (Via interface to third pary app)
Predictive Apriori	✓	
Tertius	✓	
Generalized sequential patterns	✓	
Eclat		✓ (Via interface to third pary app)

表 6-3 Evaluation

Fature	Weka	R
Prediction accuracy	✓	✓
Confusion matrix	✓	✓
AUC	✓	✓
Information-retrieval stats	✓	✓
Information-theoretic stats	✓	✓
ROC/lift charts	✓	
experiment facility		✓

表 6-4 Deployment

Fature	Weka	R
Serialized java object	✓	
Java source code(limited)	✓	
PMML(limited)		✓

6.4 基于数据挖掘和机器学习的木马检测系统

随着计算机网络技术的快速普及和发展,网络已日益成为人们日常生活和工作必不可少的一部分。但是由于计算机网络本身所具有的开放性,自2007年来,网页木马已成为网络安全攻击的最主要手段之一,据卡巴斯基、瑞星、金山等安全公司公开的数据表明,网页挂马传播通道占到计算机病毒传播的70%左右。另外,据金山毒霸安全实验室统计,每天有数百万次浏览与挂马网页有关,中毒概率在20%~50%之间。因此可以这样说,解决了网页挂马问题,上网中毒的可能性就将减少一大半。中国是网页木马的重灾区,据不完全统计,2009年中国被境外控制的计算机IP地址达100多万个;被黑客篡改的网站达4.2万个,根据360官方网站报道,BBS论坛、团购网站、高校各部门网站以及政府官方网站这三类安全性存在着大量安全漏洞,经常会遭受各类黑客攻击。2012年,Google公司警告称有超过两万个网站被JavaScript重定向恶意软件黑掉或者注入。该公司称网站的一些网页被感染,谷歌搜索质量组称JavaScript通过第三方web软件被注入到站点中,可以将访问者重定向到恶意网站。

网页木马本质上说是一个Web页,但是它又和普通的Web页有很大的区别。网页木马一般隐藏在正常的Web页中,主要利用脚本来传播木马。这些特殊网页中通常是将木马程序的执行代码编码成为网页的组成部分,并配合特殊网页代码来激活木马程序执行,因此被称为网页木马。网页木马和一般计算机病毒有很大区别。计算机病毒主要以破坏数据、破坏系统软硬件、自我复制并感染其他文件为目的;而木马则主要以偷窃数据、篡改数据为目的,一般不进行自我复制感染其他文件。危害性较大的木马一般由服务器和客户端组成,用户一旦感染上木马后,木马便会立即与服务器端进行连接,此时黑客就可以控制用户的计算机系统,这就有可能造成计算机系统被恶意破坏(如硬盘被格式化等)而无法正常运行、用户的个人信息被恶意盗取等安全问题。

网页木马(Webpage Trojan)已经成为Web安全的首要威胁,如何有效地检测网页木马已经成为信息安全的一个重要研究方向。当前网页木马主要采用脚本语言编写,因此面向网络脚本的各类木马攻击也越来越严重,如何有效

检测网站是否被植入恶意木马是网站管理者和用户都需要关心的问题。

因此,有必要为有关安全监测部门专门设计一种有效便捷的网页木马检测工具以维护网络的安全。

6.4.1 网页木马概述

1. 网页木马的定义

木马是一个大部分人已经比较熟悉的名字,但是网页木马(Web Trojan)确是一个相对比较新颖的概念。从本质上说它不属于木马类,而是一种通过网络传播恶意代码的手段。网页木马本质上说是一个 Web 页,可以是一个静态的 Html 页面,但是它与普通的 Web 网页又有很大的区别。网页木马一般会隐藏在正常的 Web 页中,利用 JavaScript、VBScript 等脚本来传播木马或者恶意的重定向 URL。这些特殊网页中通常是将木马程序的执行代码编码成为网页的组成部分,并配合特殊网页代码来激活木马程序执行,因此被称为网页木马。网页木马出现的历史要比木马和病毒的历史早得多。先前大多数以恶意网页为主,比如修改注册表、脚本死循环等,在 2001 年的时候出现过几个利用 MIME 头漏洞、BMP 网页木马等。2004 年网页木马开始流行起来,并且木马的传播手段更为多样化,从 CHM 网页木马、Flash 网页木马、WMV 网页木马、隐藏在多媒体文件的 RMVB 网页木马等多种形式。另外挂马的手段也出现了通过网页被动传播、ARP 欺骗挂马、通过 QQ 尾巴诱使用户点击链接等。

2. 网页木马的传播方式

网页木马都是一些脚本程序,利用加密或者变形的混淆方式很容易躲过许多杀毒软件的查杀。网页木马利用了系统的一些第三方软件的漏洞然后悄悄执行。网页木马的出现使得现在的网络安全形势更为严峻,把恶意脚本嵌入页面框架或者通过其他手段插入到正常页面的行为,这就是俗称的“挂马”。网页木马存在的形式日新月异,其传播手段也从修改页面到电子邮件、即世家通讯传播、ARP 欺骗传播等。根据传播方式的不同,网页木马可分为被动和主动两种方式。被动木马是指攻击者处于某种不法目的入侵某些大型网站,例如有些黑客为了偷取游戏装备或金钱。主动挂马主要是指黑客的主动攻击,例如通过发送邮件、通信软件,由认为的传播一些木马链接,诱使用户点击以达到其不法的目的。如今将木马上传到网站上已变得更加困难,因此主动挂马传播已经成为主要的传播方式。

3. 网页木马的检测方法

随着恶意脚本成为网络安全攻击的主要手段,恶意脚本的检测技术成为安全领域重要的研究方向。目前主要有基于特征码匹配、统计与分析、机器学习等检测方法。其中应用最广泛的是基于特征码匹配的检测技术,其优点是检测速度快,特征匹配时检测准确度高,但无法有效检测到使用代码迷惑和变形技术的恶意脚本,对未知恶意脚本几乎无法预测,同时随着恶意脚本的增加,恶意脚本的特征库中的特征码数量也在急剧增加,这也会极大影响到检测效率。

(1) 恶意代码检测技术

统计与分析主要是利用统计学的方法进行综合的分析,最终判别待检测网页是否含有恶意代码,如今网页恶意脚本检测在国内外统计分析的主要模型有特征危险概率模型、判断矩阵法等。统计与分析方法在对未知及变形的恶意代码方面相较传统的基于特征匹配的检测方法有一定检测能力。近年来,基于机器学习的智能检测技术已成为研究的重点,它可以利用数据挖掘从已存在的大量代码数据中挖掘出有意义的模式。利用机器学习可以帮助归纳出已知恶意代码的类别,以此来进行相似性搜索,能够有效检测未知恶意代码。

目前,机器学习检测技术尚不是十分成熟,很多方面都处于理论分析阶段,并没有广泛应用于恶意代码检测领域。网页检测分析技术可分为静态检测技术和动态检测技术。静态分析技术一般采用直接扫描程序源代码或者利用工具对源码进行反汇编然后分析其汇编代码,提取代码中的特征码,综合分析程序的行为,根据事先设定的特征库集合进行匹配检测,最终检查出程序所具有的缺陷等问题。源代码静态分析方法有较高的误报率和漏报率的问题。动态检测技术需动态执行程序,它适用于小型程序,对于大型程序不能够全部分析到,并且无论是检测速度上还是准确度上都落后于静态检测方法,但是能够免于代码迷惑和变形,这是它最大的优点。

(2) 恶意代码混淆技术

早期的病毒由于特征码固定,容易被反病毒软件检测,为了提高隐蔽性,多病毒采用变形技术来逃避特征码检测。比如熊猫烧香病毒就有成千上万种变种,对社会造成了极大的破坏。变形技术在不改变病毒功能的前提下,改变主体代码的形态,从而使其没有固定的特征码。恶意脚本代码大都已执行了混淆处理,传统的基于特征码匹配的检测技术无法对使用了代码迷惑或变形技术的恶意脚本无能为力。通常来说,被混淆的恶意脚本一般没有固定的反混淆模

式,而且反混淆过程大都比较烦琐,所以很难编制出有效程序自动进行反混淆。早期反混淆方主要采用手工方法,手工方法显然效率太低,反混淆能力有限。如何对被混淆的代码进行反混淆也是当今恶意代码检测技术重要的研究方向。

6.4.2 网页木马检测系统需求分析

随着网页木马成为网络安全攻击的主要手段,网页木马的检测技术已成为安全领域重要的研究方向。目前主要有基于特征码匹配、统计与分析、机器学习等检测方法。其中众多的杀毒软件应用最广泛的是基于特征码匹配的检测技术,其优点是检测速度快,特征匹配时检测准确度高,但无法有效检测到使用代码迷惑和变形技术的恶意脚本,对未知恶意脚本几乎无法预测,同时随着恶意代码的增加,恶意代码的特征库中的特征码数量也在急剧增加,这也会极大影响到检测效率。对于网页木马的检测大部分的安全检测软件误报率都非常高,故寻找有效的检测方案以解决网页木马的检测问题具有非常重要的理论与现实意义。动态 Html 技术(JavaScript 等)的发展给攻击者带来了一个全新和强大的攻击技术来破坏计算机系统。恶意动态 Html 代码通常被嵌入到一个正常页面中。当一个用户浏览恶意网页便会受到感染。此外,恶意代码可以通过混淆或者变化来伪装自己,这使得检测变得更加困难。防病毒软件通常使用基于签名的方法,未必能够有效检测出经过伪装的恶意 Html 代码。因此,我们提出了利用数据挖掘和机器学习技术来实现恶意网页的检测,以提高网页木马的检测准确率。

1. 系统功能性需求

随着计算机网络的快速发展,大部分的行业都提供了 Web 网络应用,浏览器也因此成了黑客等不法分子的攻击对象。黑客通过在正常网页中插入恶意代码,用户一旦访问该网页,就可能会将恶意的软件下载至客户端,使客户端感染病毒,进而破坏系统或者盗取重要信息。网页木马虽然也包含“木马”二字,但其与传统意义上的木马有很大的不同,一般采用网络脚本语言进行编写嵌入到网页当中,通过对代码采用压缩、变换、加密、冗余等手段,降低代码可读性,隐藏代码特征,使得传统的反病毒软件不能够从变种的代码中提取出特征以规避检测。网页木马对互联网用户所产生的危害不言而喻。目前,市场上尚未出现专门针对基于 JavaScript 脚本攻击的网页木马远程检测产品。因此开发与设计一款面向 JavaScript 恶意脚本攻击的远程检测工具意义重大。

鉴于此,设计本系统的目的就是在用户正常进行网页浏览还加入一个之前网页木马检测的过程,并通过分类器进行判别是否为恶意网页木马。经过对国内外恶意代码检查方法的调研,可得到以下的实现方案,该方案网页木马系统将主要由六大功能模块组成:URL黑名单、网络爬虫、特征提取模块、JavaScript脚本编译模块、BP神经网络集成分类器模块、JavaScript样本库模块。系统用例图如图6-4所示。

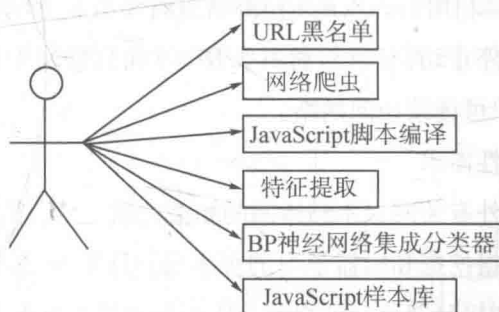


图6-4 系统用例图

网页木马检测系统的防御检测流程模型图如6-5所示。

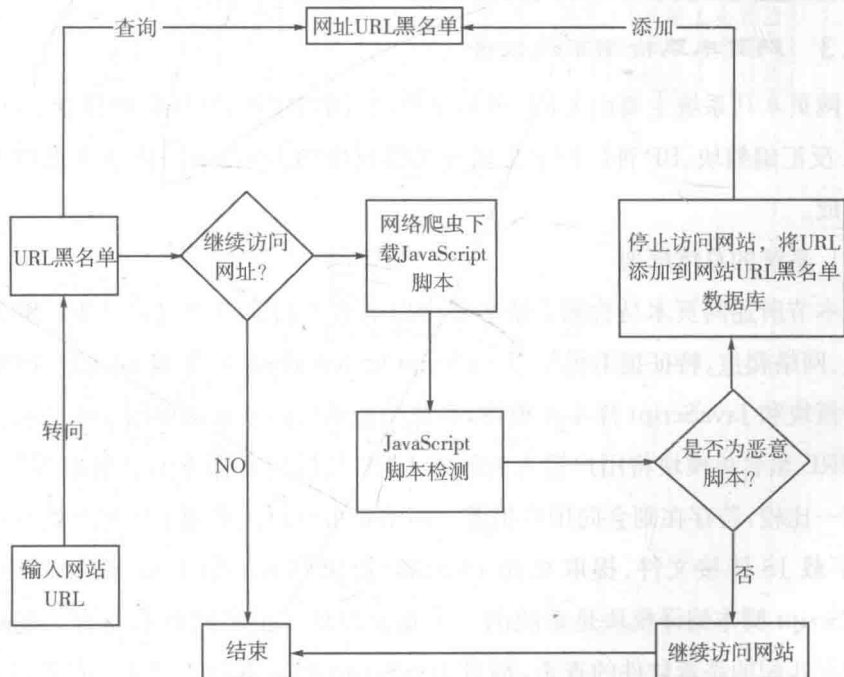


图6-5 网页木马检测流程图

(1)首先,用户输入要访问 WEB 网站的 URL 地址,系统将转向 URL 黑名单检测模块,检索黑名单 URL 数据库中是否包含发送用户输入 URL。若 URL 包含在黑名单中,系统进行提示报警,用户可自主选择是否继续访问该网站。若不访问,则检测流程结束。若用户选择继续访问,则跳转至流程(2)。

(2)系统利用网络爬虫模块下载页面中包含的 JavaScript 脚本以及 JavaScript 外部链接脚本,利用 JavaScript 检测器进行检查是否为恶意脚本,若为恶意脚本则提示用户停止访问网站,将其 URL 添加数据库 URL 黑名单中。若为正常脚本,则用户可继续访问网站。

2. 系统非功能性需求

通过查阅国内外有关网页木马检测的相关文献,分析并总结出网页木马的检测方法。基于数据挖掘和机器学习的页木马检测系统实际的工作流程包含两部分,训练部分和检测部分。训练部分主要是通过恶意脚本和正常脚本对样本数据的训练,使得网页木马分类器能够有较好的检测效果。训练完成之后得到相应的分类器模型可应用到真实的检测环境。

6.4.3 网页木马检测系统设计

网页木马系统主要由 URL 黑名单模块、网络爬虫、特征提取模块、JavaScript 反汇编模块、BP 神经网络集成分类器模块和 JavaScript 样本库模块等模块组成。

1. 系统的总体框架

本节所述网页木马检测系统主要由以下五大功能模块组成:URL 黑名单模块、网络爬虫、特征提取模块、JavaScript 脚本编译模块、集成 BP 神经网络分类器模块和 JavaScript 样本库模块,系统功能结构图示意图如图 6-6 所示。其中 URL 黑名单模块将用户输入的网站 URL 与后台数据库的黑名单 URL 进行逐一比较,若存在则会向用户报警。网络爬虫模块主要用于实现下载网页源码、下载 JS 链接文件、提取页面 JS 内容、提出网页上的 URL 链接等功能。JavaScript 脚本编译模块是系统的一个重要模块。很多网页木马为了规避基于特征匹配的杀毒软件的查杀,都对 JavaScript 脚本进行了混淆。目前,JS 混淆的应用已经非常广泛。为了去除混淆对检测的影响,系统使用 GoogleV8 引

擎对 JavaScript 脚本进行编译生成机器码。

特征提取模块对 JavaScript 编译模块生成的汇编程序使用 N-gram 特征提取技术以提取可以区分 JavaScript 脚本类型(正常或恶意)的特征。BP 神经网络集成分类器模块是系统的核心模块。特征提取模块提取的特征是一个多维度的特征向量,系统使用这些特征进行机器学习得到集成 BP 神经网络分类器模型。JS 样本库模块用于对检测后的脚本进行分类,其中脚本可分类为正常脚本、可疑脚本、恶意脚本等,补充样本库以便系统进行机器学习训练,提高检测精度。

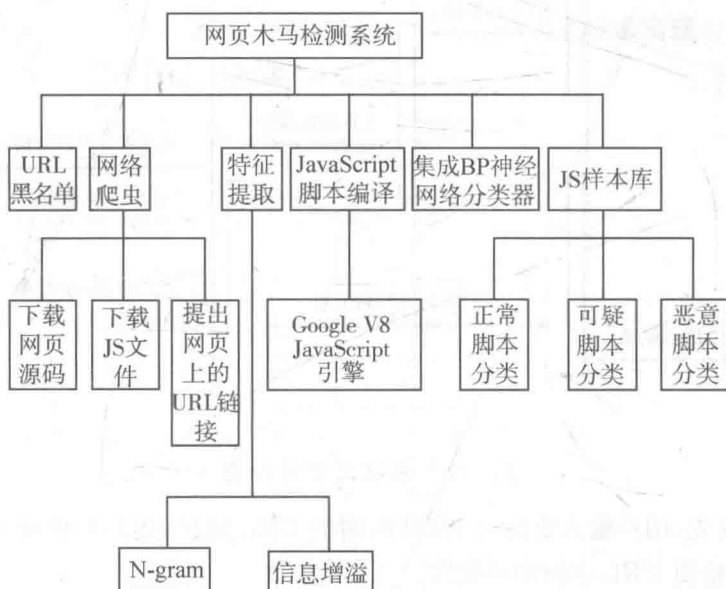


图 6-6 系统功能结构图

2. 系统工作流程

网页木马检测系统的工作时序图如 6-7 所示。

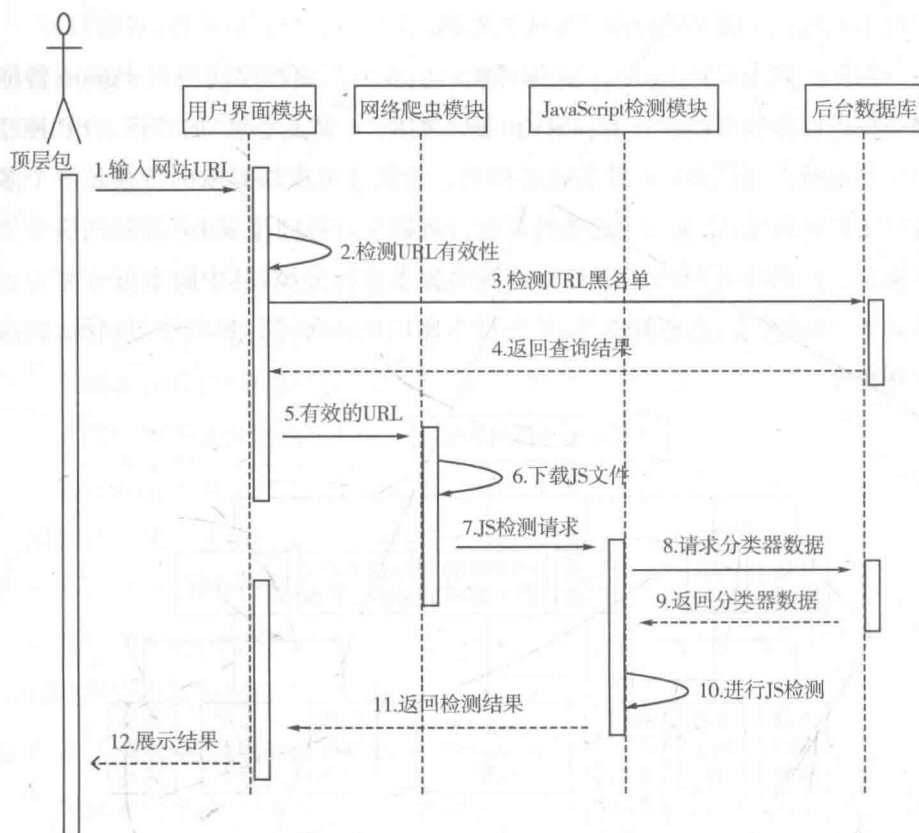


图 6-7 系统工作时序图

(1)首先,用户输入要访问 WEB 网站的 URL 地址,用户界面模块调用正则表达式检测 URL 字符串有效性。

(2)若 URL 有效通过则跳转至后台数据库进行 URL 黑名单检测模块,检索黑名单 URL 数据库中是否包含发送用户输入 URL。若 URL 包含在黑名单中,系统进行提示报警,用户可自主选择是否继续访问该网站。若不访问,则检测流程结束。若用户选择继续访问,则跳转至流程(3)。

(3)系统调用网络爬虫模块下载页面中包含的 JavaScript 脚本以及 JavaScript 外部链接脚本,向 JavaScript 检测器发送检测请求。

(4)利用 JavaScript 检测器进行检查是否为恶意脚本,若为恶意脚本则提示用户停止访问网站,将其 URL 添加数据库 URL 黑名单中。若为正常脚本,则用户可继续访问网站。

3. 系统检测流程

网页木马检测系统检测功能部分主要可分为两个部分:离线训练部分和在
线检测部分。

(1) 离线训练部分

基于机器学习的训练部分主要由数据收集和特征提取两部分组成。

首先,需要收集一些数量的正常网页(含正常的 JavaScript 脚本的网页)和
网页木马(含恶意 JavaScript 脚本的网页)作为训练的集合,利用 Google V8 脚
本引擎编译这些脚本文件,利用 N-gram 特征提取技术提取特征。

其次,选取前面出现频率最多的若干个特征作为区分正常代码和恶意代码
的有效特征码。每个训练样本根据是否包含这些特征组成一个由布尔值组成
的特征向量供分类器训练,分类器采用 BP 神经网络集成技术实现。

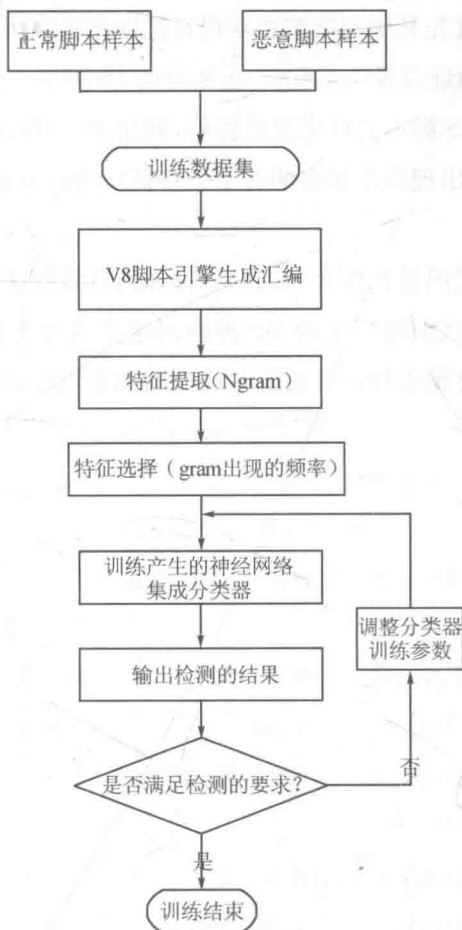


图 6-8 系统离线训练流程

系统离线训练部分流程示意图如图 6-8 所示。任何一个机器学习分类器在训练的过程中需要数据集合,训练集的大小、训练集样本是否具有代表性、样本标记是否准确都会影响分类的效果。

本文利用网络爬虫搜集了 100 个正常页面作为正常页面,又人为手动去相关网站搜集了 100 个含有恶意 JavaScript 脚本的页面,分别标记为正常脚本和恶意脚本类别,用来做训练部分的数据。每次分类器训练结束之后都会产生一个准确度,若准确度大于设定的阈值 80% 则训练结束,否则需要调整分类器的训练参数重新进行训练。

(2) 在线检测部分

在线检测部分首先利用网络爬虫获得目标网站的网页源码,从文件中提取 JS 文件外部 URL 地址以及<script>标签内的 JS 脚本。然后利用 GoogleV8 脚本引擎编译这些 JS 脚本文件生成机器码,利用 N-gram 特征提取技术提取特征,然后选取前面出现频率最多的若干个特征作为区分正常代码和恶意代码的有效特征码。

根据每个待测代码是否包含训练部分选择的有效特征,构成一个布尔向量空间,分类器采用离线训练产生的 BP 神经网络集成分类器进行检测,判断该待测代码是否为恶意脚本代码。系统在线检测部分流程示意图如图 6-9 所示。

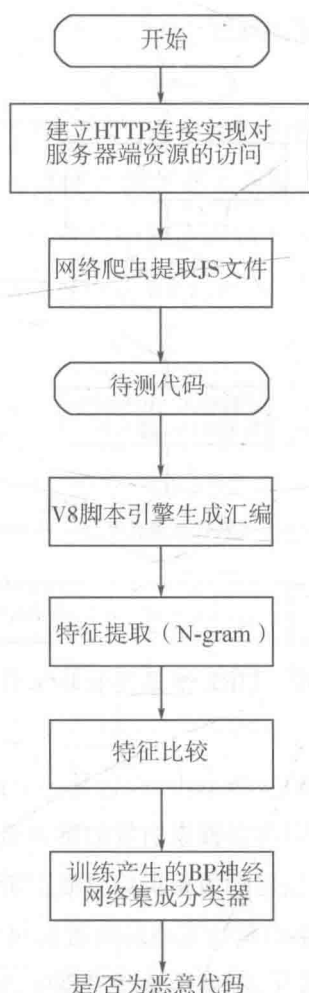


图 6-9 系统在线检测部分工作流程

4. URL 黑名单模块

用户浏览网站之前向系统输入 URL, URL 黑名单检查模块将其与后台数据库的恶意网站黑名单地址逐一进行比较, 若其包含在内系统则提示用户, 用户可自主选择是否继续访问该网站。若不访问, 则检测流程结束。其工作流程如图 6-10 所示。

用户浏览网站之前向系统输入 URL, URL 黑名单检查模块将其与后台数据库的恶意网站黑名单地址逐一进行比较, 若其包含在内系统则提示用户, 用户可自主选择是否继续访问该网站。若不访问, 则检测流程结束。其工作流程

如图 6-10 所示。

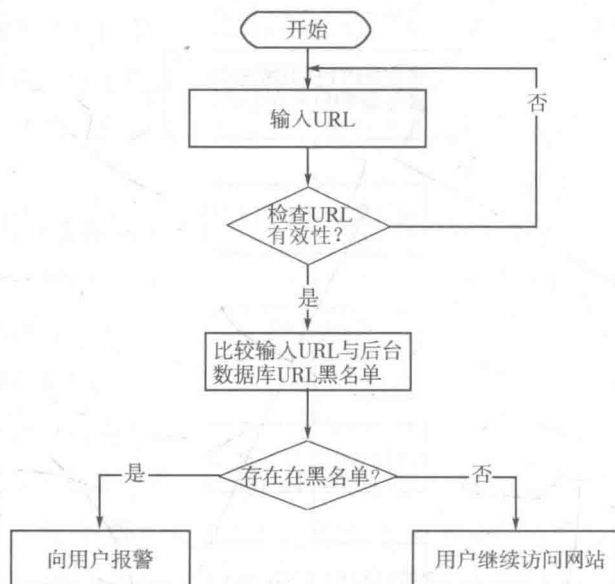


图 6-10 URL 黑名单模块工作流程

5. 网络爬虫模块设计

网络爬虫又称网络蜘蛛(web spider),它是一个能从网络上下载网页并且能够自动搜集网页内容的程序,是搜索引擎的重要组成部分。网络爬虫在抓取网页的过程中,它能够根据给定的策略识别页面上所有的超链接并将它们添加到 URL 访问列表,网络爬虫的爬行策略影响着爬虫的实现效率。目前流行的爬虫策略有广度搜索爬虫程序、深度搜索爬虫程序、定义爬行爬虫程序等。

本节所述网络爬虫模块它采用多线程机制实现,实现特定主题内容的下载。它的主要功能是下载网页源码、抓取 JS 外部链接文件、提出网页上的 URL 链接、提取<script>标签内容等。网络爬虫模块流程图如图 6-11 所示。

网络爬虫工作流程如下:

- (1)使用 WinInet 建立连接,实现对服务器端的访问;
- (2)利用 http 协议将 Web 服务器上站点的网页代码提取出来;
- (3)提取系统所需的信息,包括页面上的 URL 链接、JS 文件外部链接;
- (4)JS 文件 URL 链接入队列;
- (5)对每个 JS 文件 URL 分别启动一个用户线程下载 JS 文件;

(6)若JS文件URL队列若为空,则爬虫结束;若不为空,则跳转到(5)一直,进行队列为空。

在搜索过程之中的一个难点是页面URL链接的提取以及<script>标签内容提取等功能。我们通过创建正则表达式对象,提取匹配的内容。提取<script></script>标签内的内容正则表达式如下所示:(<[*]>[\\s\\S]*?<[/s*/script/\\s*/script/\\s"/>\\s*/script[^>]*>[\\s\\S]*?<[/s*/script/\\s*/script/\\s*>")

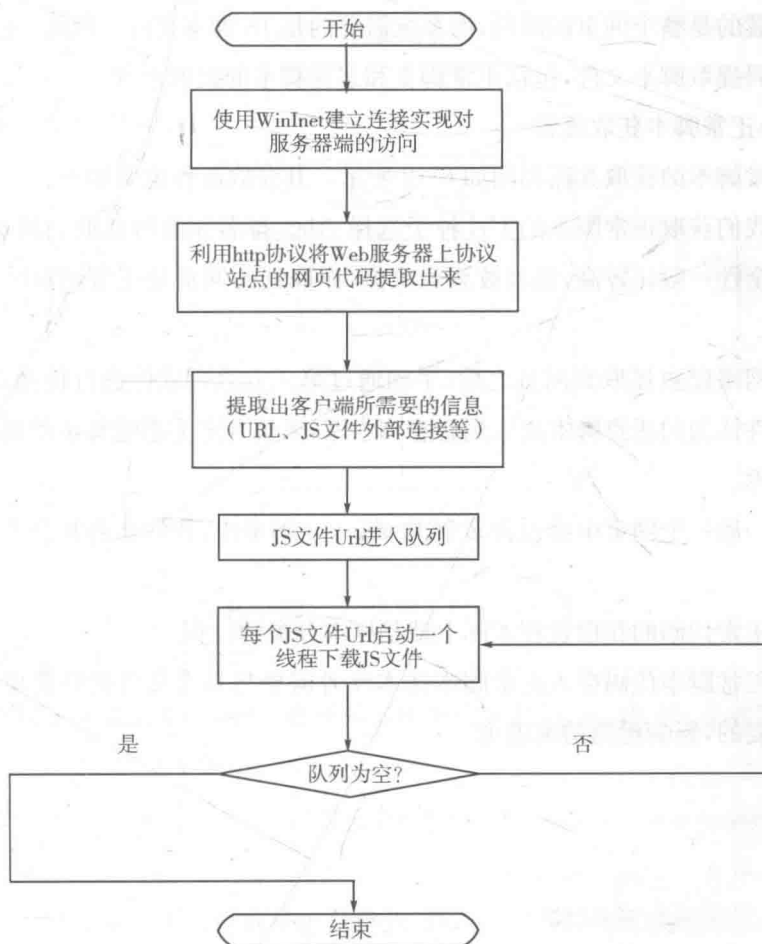


图 6-11 网络爬虫工作流程

6. JavaScript 样本收集

JS 样本收集模块的作用是建立 JavaScript 脚本样本库,不管恶意脚本检测是采用静态、动态分析方法还是采用基于机器学习分析方法,都需要大量的正常或者恶意脚本作为提取特征码的样本,为提取恶意脚本的各种特征作前期准备。

JS 样本库它可对已抓取的 JS 脚本进行分类存放,脚本可分为正常、可疑、恶意脚本三种,用户可随时查看已分类的脚本,系统可对这些脚本进行一些检测前的分析,包括脚本是否包含 eval 或 document.write 等危险函数等。网页爬虫下载的是整个网页的源码,而系统需要的是 JS 脚本文件。因此,还需要从网页源码提取脚本文件,包括正常脚本和恶意脚本的提取过程。

(1) 正常脚本获取流程

正常脚本的获取其流程图如 6-12 所示。其获取流程说明如下。

①我们获取正常脚本的 Url 种子选择 Alex 排名靠前待抓取的网页,这些网站安全性一般比较高,很难被恶意攻击,抓取到的网页是正常网页的可能性比较高。

②网络爬虫抓取到网页之后,手动通过第三方杀毒软件进行检测,将那些杀毒软件认为的恶意脚本放入到恶意脚本库,未被判定为恶意脚本的则放入正常脚本库。

③一般一个网页中会包含多个 JavaScript 脚本段,我们会将其合并到一个文件。

④正常代码的获取过程实际上就是样本分类的过程。

⑤正常脚本代码存入正常脚本样本库时需要与其他文件进行简单比对以防有重复的,影响检测的准确度。

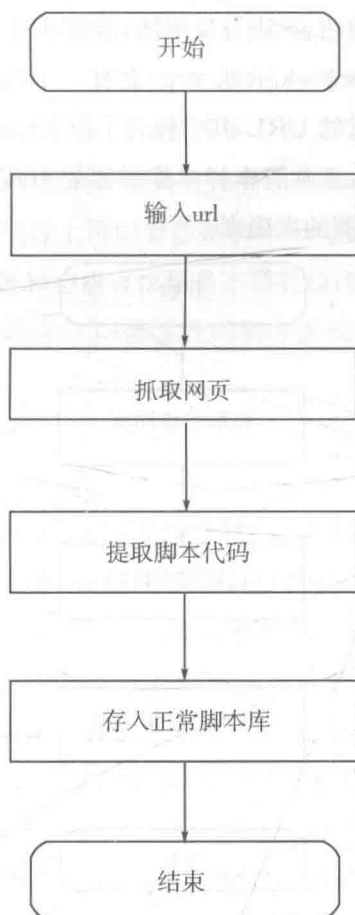


图 6-12 正常脚本的获取流程

(2) 恶意脚本获取流程

相较于正常脚本的获取,恶意脚本样本的获取则较为复杂,流程图如 6-13 所示,其获取方式如下说明。

①人工去一些反病毒网站收集网页木马,例如 <http://www.malwareurl.com> 等。

手动通过第三方杀毒软件进行检测,将那些杀毒软件认为的恶意脚本放入到恶意脚本库,未被判定为恶意脚本则放入正常脚本库。

②jsunpack-n 是一种基于 SpiderMonkey 脚本引擎的开源恶意代码监测工具,可以模拟浏览器运行检测针对浏览器和浏览器插件的 JavaScript 漏洞攻

击。它也可以对一些被混淆 JS 进行反混淆,能够通过 yara 规则库检测一些恶意脚本。网站 <http://jsunpack.jeek.org/> 上有一个网页版本的 jsunpack-n, 上面有大量已经被检测过的 URL,其中包含了很多已经检测到的恶意脚本。

③恶意脚本代码存入正常脚本样本库时需要与其他文件进行简单比对以防有重复的脚本,影响检测的准确度。



图 6-13 恶意脚本的获取流程

7. JavaScript 编译模块设计

(1) JavaScript 混淆

当前很大一部分网页木马都基于 JavaScript 脚本的形式,网页木马针对 JavaScript 出现了各种各样的隐蔽的变形手段。混淆是指被检测 JavaScript 脚本使用压缩、变换、加密、冗余等手段,降低代码可读性,隐藏代码特征。许多安全检测系统为了抵御恶意脚本的攻击,都采用基于特征匹配的方法进行 JavaScript 脚本检测。现如今攻击者为了避开安全系统的检测,采用了各种代码迷惑(混淆)、加密等变形技术,常规基于匹配的检测方法对于变形的代码无能为力,因此恶意代码变得更加难以检测。

(2) GoogleV8 脚本编译模块

GoogleV8 与 Mozilla 的 Spider Monkey 项目一样,是一个开源 JavaScript 独立引擎,可嵌入到任何 C++ 程序之中,甚至是手机应用。它在设计之初就以高效地执行大型的 JavaScript 应用程序为目的。V8 对 JavaScript 的解析不是基于反复 loop 源代码进行解释而是直接将 JavaScript 代码编译成机器码运行。换句话说,V8 引擎实际上可以看作是 JavaScript 的扩展和编译器,而传统上类似于 JavaScript 的解释型语言恰恰是不需要编译器的。本项目在对 V8 源码基础上对其进行修改以符合 JS 脚本代码编译而成的机器码可进行 n -gram 特征提取。

8. 特征提取模块设计

(1) n -gram 特征提取技术

n -gram 在计算机语言学和概率学领域中,是文本或语音给定的一种长度为 n 的持续性序列。根据不同的应用领域,该序列可以是基于字符的(Character)也可以是基于字的(Word)。 n -gram 分析已经成功应用于语言模型和语音识别。文本分类的 N -gram 分析已经成功应用于属性分类。

n -gram 是语音识别中一种语言模型。 n -gram 分析采用概率统计方法从 N -gram 集中挖掘出隐含其间的特征,将固定长度为 n 的滑动窗口在数据集的线性比特流滑动记录每个 gram 出现次数即可得到 n -gram 的统计频率分布。 n -gram 是由一个长度为 n 的滑动窗口收集的一系列重叠的子字符串,这个窗口每次滑动一个单位长度。比如,010573ee13b2,其对应的 3-gram 为 (010573), (0573ee), (73eeb2), (eeb2b2)。在 1994 年,梁婕等提出了使用一种基于字节 n -gram 的思想从恶意代码中自动提取病毒特征。 n -gram 可以捕获到一些潜在的其他方法很难准确提取的特征,在恶意代码检测领域, n -gram 是广泛应用的特征提取方法。在恶意代码中包含的一些指令是正常文件所没有或很少出现的,同样,在正常代码中也存在能够与恶意代码进行区分的指令,这些指令被称为特征。给定包含恶意代码和正常代码的数据集合, n -gram 分析方法可以从其中提取出最频繁发生的 n -grams。 n -grams 可以作为特征。当要分析一个新的代码是恶意代码还是正常代码,可通过判定该代码包含 n -grams 最多即为响应的类别。因此, n -grams 方法可以预测未知的恶意代码并可捕获使用先前学习过特征新的病毒。尽管捕获的特征隐含在提取

的 n -grams 中,但是这足够让病毒作者难以编写可以混淆 n -gram 的分析方法即使病毒作者完全了解了检测病毒的算法。

(2) 特征选择技术

利用 n -gram 方法获取的特征维数较高,因此必须将高维特征空间变换为低维的特征空间信息。增益 (Information Gain, IG) 反映了一个特征在分类中的重要性,它可以作为选择特征重要性的依据。先计算各个特征的信息增益值,将特征按信息增益值大小排序,然后忽略信息增益值小的特征以降低特征空间维数。特征选择可使用两种方法:第一种方法是直接统计每个 gram 出现的次数,这是最直接且简便的方法,但没有考虑特征与特征之间的关系以及特征在分类中的重要性。

第二种方法是使用信息增益的方法,通过计算每个特征的信息增益值在分类中的重要性,值越大(小)说明特征在分类中的重要性也越大(小),因此可以忽略增益值小的 gram 便可降低特征空间的维度。在信息论中一般用熵作为不确定性的度量。设 $X = (x_1, x_2, \dots, x_n)$, 则变量 X 的熵定义为

$$H(X) = -\sum_i p(x_i) \log_2(p(x_i)) \quad (6-1)$$

X 由另一个变量 Y 观察后的熵定义为

$$H(X|Y) = -\sum_j p(y_j) \sum_i p(x_i|y_i) \log_2(p(x_i|y_i)) \quad (6-2)$$

公式(6-2)中, $p(x)$ 是变量 X 的各个分量的先验概率, $p(y_j)$ 是变量 Y 的各个分量的先验概率, $p(x_i|y_i)$ 为 x_i 关于 y_j 的后验概率。则信息增益定义为

$$IG(X|Y) = H(X) - H(X|Y) \quad (6-3)$$

公式(6-3)说明由于 Y 的出现映射出 X 的部分信息从而导致 X 的信息熵下降。

根据步骤 JS 脚本编译实际特征产生情况,从汇编码中提取操作码来进行 n -gram 特征提取产生的不同特征数目并不多,我们采用第一种方法将最频繁出现的若干个特征作为区分正常脚本和恶意脚本的特征向量。

(3) n -gram 特征提取过程

表 6-5 所示的文件内容是一个从网络上收集的恶意 JavaScript 脚本样本。以该脚本为例子说明 n -gram 特征提取的全过程。

表 6-5 一个恶意 JavaScript 的脚本样本

```
function killErrors(){
    return true;
}

window.onerror=killErrors;

eval(function(p,a,c,k,e,d){e=function(c){return(c<a?"":e(parseInt(c/a)))+(c=c%a)>35?String.fromCharCode(c+29):c.toString(36))};if(!''.replace(/\w+/g,function(c){return d[e(c)];k=[function(e){return d[e]]};e=function(){return '\\w+'};c=1;};while(c--){if(k[c])p=p.replace(new RegExp('\\b'+e(c)+'\\b','g'),k[c]);return p;}}('6(b("c%e%2%4%0%5%d%0%a.7%2%8%9//l.k/m.o%3%1%n%g.f%3%1%h%4%j%0%5%i"))';25,25,'0D|2C|28|22|29|0A|eval|DloadDS|22http|3A|0Acom|unescape|function|7B|20DowndownloadCalcAndRun|exe|22baidu|200|7D|3B|com|268ip|baidu|20|cab'.split('|'),0,{})))

DowndownloadCalcAndRun()
```

利用 GoogleV8 JS 引擎对样本进行编译生成表 6-6 所示的机器码,限于篇幅只列出部分机器码。表中总共两列,第一列表示恶意脚本对应的机器码,第二列表示机器码对应的汇编指令。汇编指令由操作码与操作数组成,我们从汇编指令中提取出操作码然后再进行基于字 n -gram 的频率出现统计。

表 6-6 恶意脚本样本编译生成的机器码

55	pushebp
8bec	movebp,esp
56	pushesi
57	pushedi
b835011000	moveax,00100135
50	pusheax
56	pushesi
68edbd0e00	push0xebded
6a00	push0x0
e8e808ffff	call00143300
3b25006a4301	cmpesp,[0x1436a00]
0f82fb000000	jc287(00152B1F)
8b4617	moveax,[esi+0x17]
50	pusheax
b925541000	movecx,00105425
e88ee5feff	callLoadIC_Initialize(00140FC0)

利用 windows 批处理或 linuxshell 脚本的文本处理功能,提取出表 6-6 的汇编程序指令对应的操作码生成表 6-7 对应的临时文件。

表 6-7 包含操作码的临时文件

```

push
mov
push
push
mov
push
push
push
push
call

```

n -gram 中 n 取值 2, 采用基于字的 2-gram 统计每个 gram 出现的次数, 最终的统计结果如表 6-8 所示, 如 mov_mov 特征统计出现的次数为 1642 次。

表 6-8 基于字的 2-gram 统计

```

mov_mov1642
push_push850
mov_push556
jmp_mov458
code_embedded456
mov_jmp450
mov_call436
position_code436
embedded_embedded428
embedded_statement422
push_mov414
push_call378
call_mov348

```

(4)n-gram 类图

类图(ClassDiagram)是面向对象系统 UML 建模中最常用的图,是定义其他图的基础。类图主要是用来显示系统中的类、接口以及它们之间的静态结构和关系的一种静态模型。类的关系有泛化(Generalization)、实现(Realization)、依赖(Dependency)和关联(Association)。其中关联又分为一般关联关系和聚合关系(Aggregation),合成关系(Composition)。n-gram 特征提取技术类图如 6-14 所示。

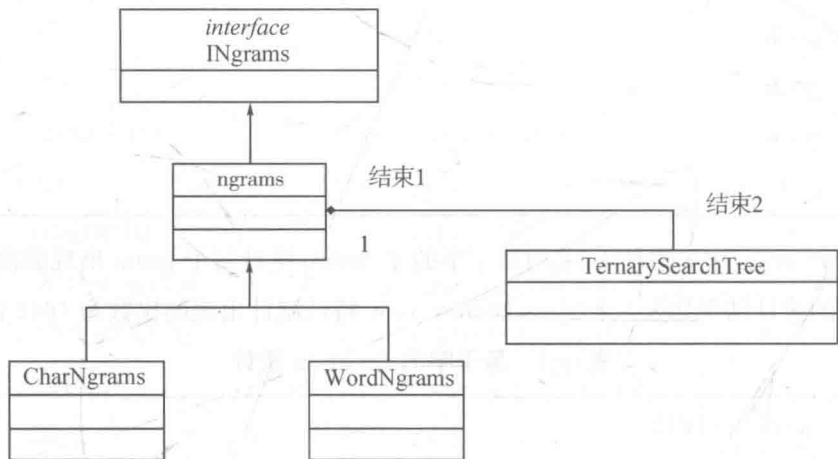


图 6-14 n-gram 类图

9. 集成分类器模块设计

(1)BP 神经网络

在计算机科学和相关领域,人工神经网络(Artificial Neural Network)计算模型的灵感主要是来自动物的中枢神经系统(特别是大脑),使其具有机器学习和模式识别的能力。他们通常表现为相互关联的“神经元”的系统,向网络中输入信息值可以计算出对应的值。ANN 是一个计算模型,它包含三个方面:定义神经网络数据结构的神经网络图;指明学习将如何进行的学习算法;确定网络的传播过程。

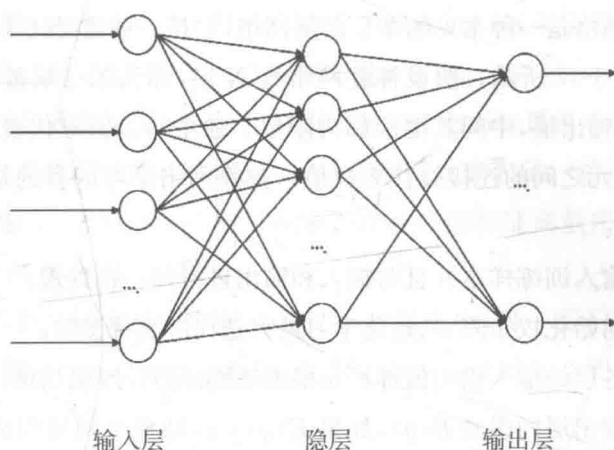


图 6-15 BP 神经网络基本结构图

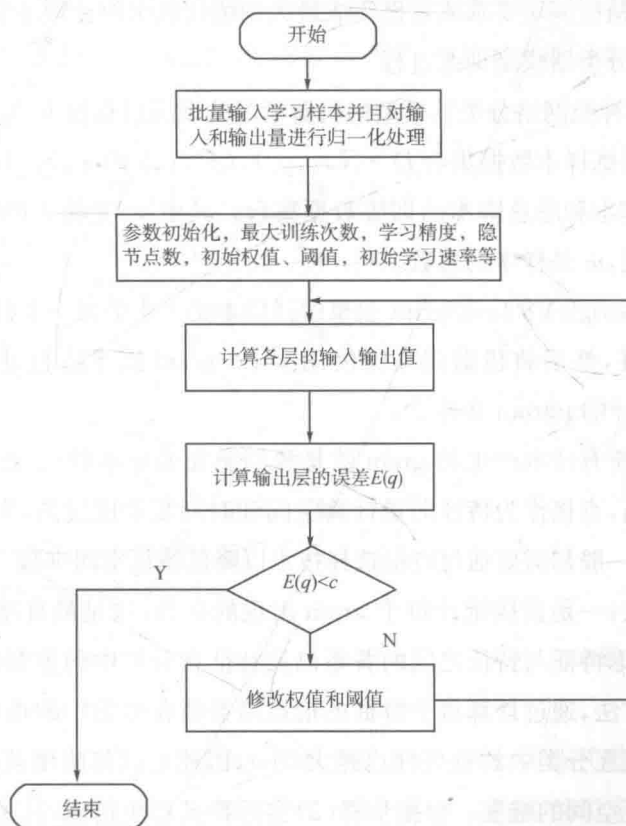


图 6-16 BP 神经网络算法流程

BP 神经网络是一种常见的多层前向网络,它是一种监督式学习的算法,其结构示意图如 6-17 所示。假设神经网络有 N 层,那么第一层被称为输入层,最后一层称为输出层,中间其他层称为隐层。途中每个圆形代表一个神经元,神经元与神经元之间的连接线代表权值。神经网络学习的目的是根据给定输入的样本输出:

- (1) 批量输入训练样本并且对输入和输出进行归一化处理;
- (2) 随机初始化权矩阵,初始化学习最大迭代次数等参数;
- (3) 计算各层的输入输出值神经元根据激励函数计算输出值;
- (4) 计算输出层的误差 $E(q)$,如果 $E(q) < \epsilon$,则神经网络训练结束可输出网络各层的权值以及输出层的结构;否则,转至步骤(3)继续计算各层的输入输出值,一直到精度满足要求或者已完成最大训练代数则停止训练学习。

(2) 集成分类器模型训练过程

集成 BP 神经网络分类器模型的机器学习的训练过程如下。

① 给定训练样本数据集 $D = \{(x_1, y_1), (x_2, y_2) \cdots (x_m, y_m)\}$ 包含了一定数量的正常脚本和恶意脚本的训练数据集。其中 x_i 是输入的特征向量, y_i 是样本的类别, m 是样本的总数。

② 利用 GoogleV8 JavaScript 引擎编译样本集合中的每一个样本以产生一个机器码文件,然后将机器码文件使用字 n -gram 技术进行处理产生包含 gram 数目统计的 ngram 文件。

③ 此时,所有样本产生的 gram 就是我们需要的脚本特征,但是通常其空间维度非常高,直接作为特征向量计算空间和时间复杂度过高,实际不可能直接使用,一般都需要通过特征选择技术以降低特征空间维数。特征选择可使用两种方法:一是直接统计每个 gram 出现的次数,这是最直接且简便的方法,但没有考虑特征与特征之间的关系以及特征在分类中的重要性;二是使用信息增益的方法,通过计算每个特征的信息增益值在分类中的重要性,值越大(小)说明特征在分类中的重要性也越大(小),因此可以忽略增益值小的 gram 便可降低特征空间的维度。根据步骤(2)实际特征产生情况,从汇编码中提取操作码来进行 n -gram 特征提取产生的不同特征数目并不多,我们就将最频繁出现的 200 个 gram 特征作为区分类别的特征向量。

④特征提取和选择技术从这些样本当中提取可以区别正常脚本和恶意脚本的特征。我们将特征用一个 n 维度的向量 $F = \{f_1, f_2, f_3 \cdots f_n\}$ 进行表示。

⑤遍历每个样本的 n -gram 文件判定其每个 gram 是否有特征向量 F 包含的特征。如果包含则用 1 来表示,不包含则用 0 来表示。经过处理之后,每个样本就可以用一个只包含 0 和 1 布尔值的 m 维的特征向量来表示,如 $\{0, 1, 1 \cdots 0\}$ 表示样本包含特征 f_2 和 f_3 。

⑥将这些特征样本集中存储到文件或数据库中,每一行记录代表一个样本,最后一个数字代表样本的类别,我们用 1 表示正常脚本类别,用 -1 表示恶意脚本类别。

⑦最后,使用 AadBoost 集成 BP 神经网络算法进行分类模型的学习。首先,需要设定分类模型的各个参数,特征向量的维度 n 设定为 200,维度也可根据实际处理情况进行调整;其次,样本数 m 为 200,其中正常脚本和恶意脚本数都各为 100。

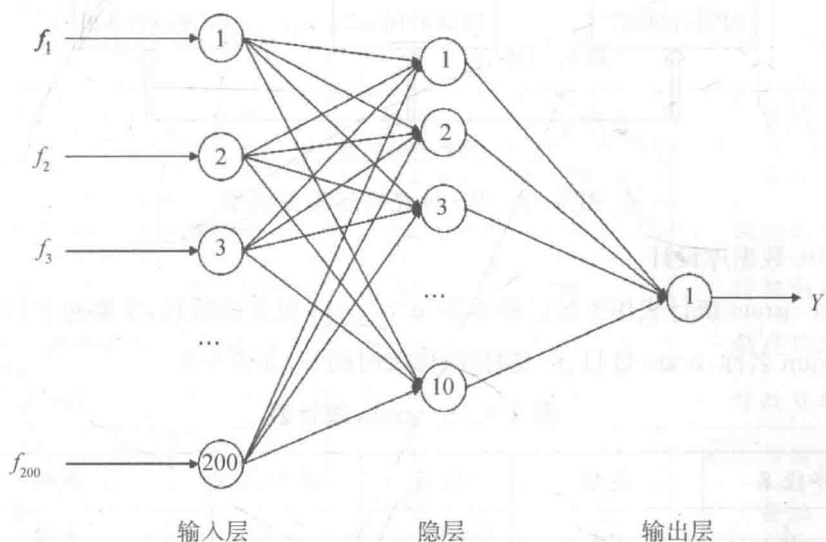


图 6-17 单个 BP 神经网络模型

⑧单个学习器 BP 神经网络的参数设置如下:最大训练代数设置为 1000,学习精度设置为 $1e-3$,隐节点数设置为 10,初始权值范围设定为 $(-0.5, 0.5)$ 。如何合理地选取 BP 神经网络的隐层数和隐层包含的节点数目,这在理论上并没有给出行之有效的方法,往往需要根据实际问题进行分析。本系统在具

体设计时,通过对不同神经元数进行,训练对比设定隐节点数目为 10,训练结果的精度能够提高。单个 BP 神经网络模型拓扑结构包括输入层、隐层和输出层。其中输入层 f_i 表示单个特征样本向本的一个属性, Y 表示样本的输出(即样本所属的类别)。

⑨图 6-18 展示的是单个弱学习模型的基本结构。BP 神经网络集成是用有限个如图 6-17 所示的 BP 神经网络对样本数据集进行学习,其结构如图6-18 所示,我们设置基本的学习器数目为 20。

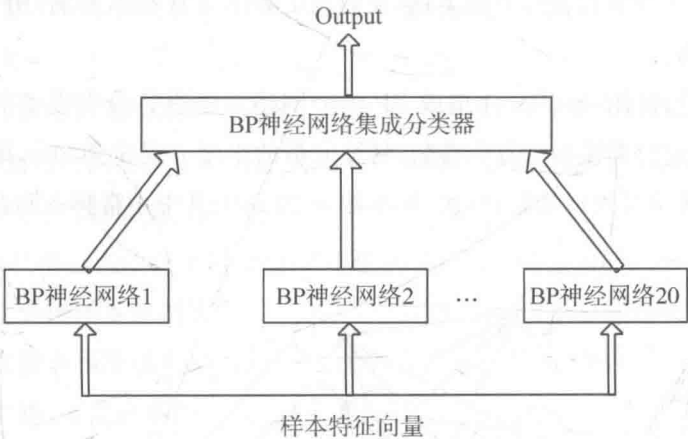


图 6-18 BP 神经网络集成模型

10. 数据库设计

n-gram 统计表用于统计样本集每个 gram 包含的数目,主要包含以下信息:gram 名称、gram 数目、创建时间、修改时间等,如表 6-9。

表 6-9 n-gram 统计表

字段名	类型	长度	是否为空	说明
id	int	11	N	主键
gram	varchar	50	N	gram 字符特征
num	int	11	N	gram 数量
createDate	datetime	0	N	创建日期
modifyDate	datetime	0	N	修改日期

网站 URL 黑名单表结构如表 6-10 所示,其主要包括网站名称、网站 URL、创建时间、修改时间、备注等信息。

表 6-10 网站黑名单 URL 表

字段名	类型	长度	是否为空	说明
id	int	11	N	主键
Name	varchar	100	N	网站名称
url	varchar	500	N	URL 地址
createDate	datetime	0	Y	创建日期
modifyDate	datetime	0	Y	修改日期
remark	varchar	255	Y	备注

n-gram 样本特征表主要包含样本的特征布尔向量、样本的类型等信息,如表 6-11。

表 6-11 n-gram 样本特征表

字段名	类型	长度	是否为空	说明
id	int	11	N	主键
filename	varchar	100	N	网站名称
Feature	varchar	500	Y	特征向量
createDate	datetime	0	Y	创建日期
modifyDate	datetime	0	Y	修改日期
class	int	11	N	样本所述类别
remark	varchar	255	Y	备注

JavaScript 文件索引表,每个网站对应一个 JavaScript 文件索引表,如表 6-12。

表 6-12 JavaScript 文件索引表

字段名	类型	长度	是否为空	说明
id	int	11	N	主键
Jsuel	varchar	100	N	JS 文件地址
IndexTableid	int	500	N	网站索引表 id
jsfilesize	int	11	N	JS 文件大小
jsdilenumber	int	11	N	JS 文件编号
createDate	datetime	0	Y	创建日期
modifyDate	datetime	0	Y	修改日期
Isobfuscated	int	11	N	JS 是否混淆
kind	varcha	255	Y	JS 种类

网站文件索引表,每个网站对应一个 JavaScript 文件索引表,如表 6-13。

表 6-13 网站文件索引表

字段名	类型	长度	是否为空	说明
id	int	11	N	主键
url	varchar	100	N	JS 文件地址
dir	varchar	500	N	目录
Creatdate	datetime	0	Y	创建日期
modifyeDate	datetime	0	Y	修改日期
Remark	varchar	255	N	备注

6.4.4 木马检测系统设计小结

本节在网页木马检测系统涉及的相关理论与技术的基础上对基于数据挖掘和机器学习的网页木马检测系统进行总体设计。网页木马系统主要由 URL 黑名单模块、网络爬虫、特征提取模块、JavaScript 反汇编模块、BP 神经网络集成分类器模块和 JavaScript 样本库模块等模块组成。

6.5 机器学习技术在数据挖掘中的商业应用研究

数据挖掘是数据库和信息决策领域最前沿的研究方向之一。数据挖掘能够揭示隐藏的模式和关系。从技术角度来看,数据挖掘是指从数据中提取隐含的、人们事先不知道的、但又是潜在有用的信息和知识的过程。从商业角度看,数据挖掘是按企业既定的业务目标,对大量的企业数据进行探索和分析,揭示隐藏的、未知的或验证已知的规律性,并进一步将其模型化的方法。Meta-Group 曾对数据挖掘做出这样的评论:“全球重要的企业、组织会发现,到 21 世纪数据挖掘技术将是他们商业成功与否的至关重要的影响因素。”数据挖掘是 80 年代投资人工智能研究项目失败后,人工智能转入实际应用时提出的。它是一个新兴的、面向商业应用的交叉学科。数据挖掘的主要方法可以分为两大类:统计学方法和机器学习方法。在数据挖掘领域,机器学习方法以其强大的处理不同类型数据的能力和商业应用的巨大潜力日益受到该领域学术界和商业界的重视。

6.5.1 技术制造商案例研究

生产线是一个粒度分析和大数据相结合的地方。能够迅速确定某个设备是否正常运作,隔离掉有缺陷的零件、错位的机器或者表现不好的流程对于结果是至关重要的。仅在 2013 年,就有数以亿计的电视机、手机、平板电脑和其他电子设备被生产出来,预测在今后几年这个数量还会继续增长。

1. 发现设备缺陷

访问最先进、最大的电子设备生产商的生产设备几乎像是跨入了一个不同的世界:人们的穿着就像太空服,机器人完美地同步动作,随着机器轰鸣,产品经过许多步骤从原材料变成零件,然后最终成为一个经过检测的无瑕疵的成品。

当在生产过程中察觉出错误,就需要立刻识别出问题的来源、其潜在影响以及如何对其进行修正。完成这些工作需要分析上千个可能的变量以及上百万个乃至数十亿个数据点。这个数据的规模非常大,而且完成分析所要求的时间也很严苛。在发现问题后只有有限的时间窗口来阻止有缺陷的产品批次留在工厂中,同时也在更多的时间和原料被浪费之前解决掉根本原因。对瑕疵品的有效处理有助于维持良好的品牌声誉、员工士气和一个更好的基线水平。

2. 如何降低成本

如何降低成本,这类建模问题的典型解决路径被归纳为下面六个步骤。

(1)第一步,产品的检查流程。为了确保满足质量控制指导方针,保证产品无缺陷,每个产品都需要经过一个规定的检查流程。例如,确定某个批次的硅片是否达到质量控制指标与确定某个计算机显示器是否能通过质量控制验收是不同的。在硅片的例子中,结果主要是一个二元状态:硅片能工作或者不能工作。如果它不工作,那它就是个没用的废弃物,因为没有有一个机械装置能修复它。在计算机显示器的例子中,显示器未达到质量控制指标可能因为存在着一个坏像素点、组装异常、无法上色或者其他数十种原因。在这些产品缺陷中,有一些可以通过重复组装过程或换掉有瑕疵的零件来修正。其他一些原因不能被修正,则该显示器就必须被当成废品销毁。对于这些复杂的设备,比起教会机器看,肉眼始终在确定显示器是否有瑕疵的目检中表现更好。肉眼在查看模式和注意到异常方面非常先进。在这个生产环境中,人工质量控制团队成员可以发现更多被视为瑕疵品的原因,比计算机系统多出一个数量级。

如果某个产品批次不能达到质量控制门槛,那么就开始进行调查。因为调查会延缓或停止生产更多产品,所以调查并不是随机进行,而是只在一个批次的产品不能通过质量控制检查后才进行。这个调查就所耗费的时间、产量的减少和对生产的干扰来看,其成本是非常昂贵的,但是它对于防止对故障产品的组装而言至关重要。

(2)第二步,假设有一批次的硅片不能通过质量控制,那么就要对企业数据仓库(EDW)提出数据提取需求。这个数据是来自许多不同来源系统的过去几天中的传感器数据。一些数据存储在传统的大规模并行处理(MPP)数据库中,其他数据存储 in Hadoop 中,还有其他一些数据既不在 Hadoop 也不在 MPP 数据库里。这是一个复杂的过程,包括将存储在第三方格式的关系数据库中的许多表进行合并,将其去标准化,转换为矩阵数据表,并使其与 MPP 中的表相匹配。生产一个硅片的过程包括几千个步骤,并带来伴随着生产过程的超过两万个数据点。当这个数据被转化,与每天生产的上百万硅片相结合,你就有了真正的大数据。这个集成在一起的矩阵数据表所达到的规模将超过物理能力,无法在规定的几个小时的时间窗口内处理完毕。

(3)第三步,将数据聚类。对于这样的宽数据(超过 1 万列),很明显并不是

所有的数据都会是导致缺陷的原因。我们可以基于许多标准去除掉一系列或将列聚类。首先,如果这个列是一元的,或者只有一个数值,就可以将它剔除掉。当某个列只有一个数据值,该列的信息就不能用来确定出现缺陷的原因。其次,是多重共线性。多重共线性是一个统计学概念,它是指有两个或更多个输入变量(在这个例子中是传感器的测量值)为高度相关。多重共线性不响应模型的最终预测能力,但是它会不必要地增加建模的计算成本,因为模型考虑的项超过了它真正需要的数量。多重共线性会导致参数估计受到其他输入变量的严重影响并且更加不可靠。这个聚类的步骤去掉了无用的变量和那些由于与其他输入变量具有相关关系而不能提高模型预测能力的变量。

(4)第四步,执行设备路径分析,努力识别问题的可能来源。这个分析可以用许多技术来完成,但是决策树通常是最常见的选择,因为它具有很好的解释性和简单的假设。过去,在所需要的时间窗口中应用所必需的数据规模是一个困难的问题。

决策树的第一层试图通过评估所有候选变量和所有候选分割点,将一套数据分割成两个最纯粹的组。最佳分割的运算过程必须被传递、归总和传播(在分布式环境的情况下),完成这些之后才能开始第二个分割点的运算。在一个并行和分布式环境中,由于在每个步骤之后都要进行沟通,需要特别关注能够高效构建决策树的技术。

当找到最佳分割点之后,必须对它进行评估,看这个分割能否增加这棵树的总体解释能力。几乎在建立每个决策树的过程中,算法都会达到这样的一些点,在这个点上,更多的分割不会优化结果反而会不必要地使这个树形模型更加复杂。这违背了简约性原则。简约性原则是指如果两个模型(在这个例子中是决策树)有相同的解释性或预测能力,那么简单的模型更优。如果通过设备路径分析找到可疑的设备,就对它进行调查。过去,决策树被证明在揭示系统性错误(某个机器失调或者仪表不准确)方面非常有效。当造成缺陷的原因具有相互作用的时候,决策树的探查表现并不是太好。这些问题通常被称为随机事件,因为它们并不归属于某个单一原因,所以不能被简单地判断和修复。这种情况的一个例子可能是在步骤 341 中,一点微尘干扰了项目 123 的编码。这个错误编码导致在步骤 297 中项目 113 的延误,而这又导致了在硅片 A4 区的

温度过高。由于 A4 区的温度过高,步骤 632 的化学浸浴不够充分,随后这又导致了……用决策树是看不出这种类型的复杂反应的。所以其他针对时间序列数据的技术功能维度分析(Functional Dimensional Analysis, FDA)或者符号累积近似法(Symbolic Aggregate Approximation, SAX)——也被加以评估。这应当会提醒我们,即使已经获得了显著的改进,但为了优化流程仍然有更多的研究和工作需要进行。

(5)第五步,公司还进行了根本原因分析,旨在识别出在程序或流程中的问题,这些问题将会导致出错,这个分析还要识别出为了防止错误需要进行评估的最小常见变量集。所有这些测量指标、属性和元数据的规模如此之大,以致对每个数据都予以考虑将会耗费大量时间,极不现实。所以最小常见变量分析或关键少数分析通过一个变量重要性测度来确定为了阻止这类错误的再次发生,需要增加哪些额外检查。

(6)第六步,将来自步骤(3)和步骤(4)的结果进行排序、分级、分析其频率。当归纳出这个信息,就可以完成关于具体时间轴、问题、建议和下一步骤的报告。

在这个例子中,客户用 Hadoop 分布式文件系统及内存计算方法来解决业务参数内部的问题。过去几年中,这一流程已经被开发、提炼和优化,但是直到客户采用了最先进的分布式内存分析技术,这样数据规模的问题才会被认为是可解决的。当采用了这个新的基础设施之后,计算一个所需规模的相关矩阵的时间从几小时下降到只需要几分钟,在调查的其他分析步骤中也看到了类似的改进。客户对于他们现在具有赢得在高技术制造行业的竞争优势并进一步优化其流程、提高在市场中的地位的机会感到由衷地高兴。

6.5.2 在线品牌管理的案例研究

在当前这个信息化时代,商业成功与你在因特网上的成功紧密相连。这对于制造商来说更是千真万确。在线品牌管理处于一个充满了挑战、快速前进的环境中,过去要几周时间才能演进的观点或认知现在在几小时之内就会形成。最流行的微博网站 Twitter 报告称在 2012 年秋天,该网站上每天都能看到 5 亿条推文。对这个大型公司的快速调查结果揭示了来自其客户的上千条推文。

不分昼夜,每过5秒,就有超过20条新推文出现。对于任何人或任何部门,跟进所有客户仅仅来自这个微博服务的对公司的评论都是不可能的事情。现在加上其他流行社交媒介网站如Facebook,LinkedIn和其他网站的更新,使得跟上消费者对你产品的最新看法显得是个不可能完成的任务。可以看到,为了能对品牌价值保持精确看法以及对品牌提升、品牌维护或品牌预期变化做好准备,所要处理的信息量将会令人却步。

许多分析技术可以协助完成这一任务。这里,我们将讨论Twitter,但这6个技术可以用于几乎任何人工生成的数据源。

(1)识别用户发表评论所用的语言。语言识别是重要的第一步,因为它能帮助确定问题或赞誉的规模和范围。大体上,Twitter用户只会用一种语言使用网站,少数用户应用两种语言(几乎总是母语加上英语)。语言识别还有助于在没有GPS定位信息时识别位置。(在笔者的经验中,GPS定位信息只会带来很小的好处,因为几乎所有推文都是来自智能手机,使得在他们的元数据中就包括了位置信息。)

(2)寻找情感倾向。情感态度分析是一个管理品牌的美妙工具,因为它能帮助你聚焦于那些不满意的客户或导致了大部分问题的产品方面,或者帮助你关注到那些能对你有帮助的支持品牌的用户。情感态度分析的真正力量是它不仅让你知道客户对你的产品的感觉,而且还能告诉你他们的感觉强度如何。它还能揭示出他们是对整个产品还是仅仅是对产品的某些方面的感觉是如此。例如,考虑一个LCD电视显示器的例子。这是一个复杂的产品,除了对它的总体态度之外,还有许多喜欢或不喜欢的个体特征。通过首先识别语言,我们可以看到巴西的负面评论高于平均水平,这可能导向一个对用户手册翻译质量的调查,或者它可能指出一个产品缺陷,当电视播放几小时后(像世界杯期间),色彩饱和度发生变化,更不容易辨别电视内容的细节。

(3)内容分类。对于是在内容分类之前进行情绪分析还是反过来存在着一些争议,我认为这归结为个人偏好问题。我对这个顺序的个人偏好是先用情绪分析作为筛选出最紧急问题、最不开心的客户的工具,然后再将那些反馈项归类。

(4)对该情况应用商业法则。一旦我们识别出特定用户的情绪倾向,以及

使他们烦恼或高兴的问题类别,我们就必须用已经建立的业务流程解决其关注的问题,或在未来活动中更积极地做出反馈。如果当前业务流程不包括具体如何对这类情况做出回应的细节,那么就必须建立能解决问题的业务流程,但这是另一个要讨论的问题,不在本书中涉及。

一个对正向反馈做出回应的标准业务流程的例子是转发推文或点赞,这样关注你的用户就会看到这个积极评论,并可能提高他们的品牌忠诚度且会对你的品牌看法更加积极。在负面评论的例子中,可以选择给客户发私信,向他们发送离他们最近的零售店列表,这样他们可以去更换产品,或者发送一个能够处理这一问题的常见问题(Frequently Asked Question, FAQ)页面链接。这个动作并不需要很大,但必须有效,它应当针对客户的具体问题,而且在客户提出抱怨后能迅速回应。如果成功做到这些,你就能确保这个用户今后更加忠诚,而且该用户可能还会影响处于其影响网络中的人,使他们对品牌的感知更加积极。

(5)创建可以用来作为反馈回路的模型。我们可以将这个客户提供的具体细节信息加入已经存在的结构化数据中,生成模型来用于许多有用的目的。我们可以将积极反馈包含到销售预测更新中,并且触发供应商提供更多配件。负面反馈则可以通过对它的学习来优化产品质量、处理这与制造流程、客户服务培训或可用性设计相关的问题。在这个步骤中,我们可以从批评者而不是支持者中学到更多的东西,通常都是如此。虽然没有人想要得到负面反馈,但这通常是产品优化的最佳来源。

(6)监控流程。一旦识别情绪、评论分类、按照业务规则处理反馈的流程到位,在这个领域中就有持续改进的机会,就像其他领域一样。这个领域的关键指标是沟通发布次数、平均响应时间和问题升级的数量。寻找在品牌成功与在线投入之间的相关关系同样也会是一个有价值的努力。我怀疑在大多数情况下能否建立因果关系,但是展示出你的投入所带来的功效,会是保证获得资金支持并显示出你的工作为公司带来额外价值的最佳方式。

这个案例研究主要聚焦于你能用来监控品牌和产品系列的方式。我们还可以用同样的方法来从竞争对手那里争取市场份额。很少有比人们刚刚表达出对使用习惯(使用特定品牌是一个习惯)的不满之后的这个时间点更容易让

他改变习惯了。你能走多远,取决于你的品牌所销售的商品或服务是什么。服务比耐用品更容易被取代。但始终保持警惕,留意从竞争对手那里争夺市场份额,对于业务的繁荣发展是一个久经考验的方法。

6.5.3 移动应用推荐的案例研究

随着过去5年中智能手机的迅速发展,智能手机应用的新市场已经出现,对精神份额的竞争非常激烈。这个领域的所有供应商都希望自己有能力向用户推荐其喜欢的应用(APP)。它们的目标首先是确保推荐不会被看作是垃圾而是个好建议,然后将自己从只是提供建议的角色转化为一个受到高度赞扬的备受信任的顾问角色。在这个例子中的商业问题是非常简单的:在用户已经安装在其智能手机上的应用中,他们可能会使用哪些应用?解决这个问题面临着许多挑战。首先,是数据的规模,有上亿的手机用户,每个用户在购买应用方面几乎都是独特的,发现好的推荐非常困难;其次,是应用的详细程度,对于这个商业挑战,数据是一套二元变量集,可以用许多不同的方式来定义这个二元状态:他们是否安装了这个应用?他们是否曾经使用过这个应用?在最近时期该应用是否被使用过?

无论你怎么设计问题,输出结果都是一个基于一套二元输入变量得出的购买应用(可能为免费)的概率。

这个问题的处理被分为几个阶段,每个阶段需要不同的技能集。

第一个阶段是数据采集和归并。传统上,通过数据挖掘进行预测建模一般都要求输入数据是矩阵形式,缺失值会对一些算法带来严重问题。在这个例子中,我们有一张要推荐的潜在应用列表,它将所有可能购买的应用都作为变量,所有客户则作为数据中的行(见表6-14),从而生成了一个非常稀疏的数据集(有许多的缺失值)。通常,归责会是一个处理这些缺失值的方法,但因为数据太过稀疏且变量为二元,归因很难不产生偏差和扭曲结果。这个数据高度缺失的情况使得神经网络和 Logistic 回归这些技术对处理这类问题的效果不佳。决策树可以更好地处理缺失值,但它们会倾向于不够稳定。因子分解机或随机梯度下降技术通常在处理这些类型的问题上表现更优越。

表 6-14 稀疏的客户—产品数据表

客户	产品 1	产品 2	产品 3	产品 4		产品 N
客户 1	1					1
客户 2			1			
客户 3				1		
客户 4	1					
.....						
客户 N		1				

第二个阶段是数据规模。因为在只考虑一部分数据子集的情况下制订的推荐是次优的,推荐的质量会受到质疑。所以,必须分配充裕的计算硬件特别是主存储器以适应问题规模,同时,软件应设计为可并行,能在分布式环境下运行。这个并行的本质使得机器上的所有 CPU 核都能被应用起来,而分布式特征则允许多个机器为解决这个问题共同运作。当解决问题所需要处理的数据规模变大的时候(在成功实施之后经常发生这一情况:新的数据规模更大的问题出现了,如果它们能被解决,就会带来更大的商业价值,因此也就备受期待),这个分布式本质还能考虑到硬件的扩展性,软件的并行和分布式设计也要考虑到如何满足速度这一关键问题。如果需要在几秒内得到答案(大部分人只愿意等这么久)而且答案又不能预先计算,那么花一天时间解决问题就毫无用处。这个问题通常可以通过进行离线训练来克服。用历史数据训练和优化推荐模型,然后部署模型,这样就可以对新事件(如某个用户来到应用商店)评分,并推荐应用。这个离线训练和模型评分模式解决了快速推荐的问题,但带来了模型有效期或者说模型衰退的问题。对需要接近于实时或更快给出结果的推荐,具有快速训练、部署和重新训练的能力是个理想情形。此外,当模型被训练并保存在内存中时,可以应用基于服务器的架构。当需要给出一个推荐,就在应用程序界面(API)上录入新纪录,然后应用基于内容的过滤或者协同过滤方法制订推荐。这些方法在虚拟商店中基于标准产生了从所有可用的应用集合中得到的一个很小的子集,然后更新总表,这样未来制订推荐的时候就可以应用到

所有的信息。

在这个案例研究中考虑了大约 200 个应用。第一步是寻找有助于缩小问题范围的集群。聚类能减少输入变量的个数,将它们从 200 个下降到只有几十个。当完成聚类后,聚类变量被添加到潜在输入变量列表中。其他技术还包括奇异值分解和发现应用之间关系的主成分分析。当生成增加了补充特征的数据表后,采用变量选择技术(无论是监督式还是非监督式)去掉不能提供有用信息的变量。然后是不同树型模型的多重迭代,这些树被改进,用多种方式进行组合,形成强大而稳定的模型。用早前被区分出来的保留样本对这些模型进行评估,这些保留样本通常并非随机样本,而是来自某个时间窗的样本。随机抽样会造成推荐的偏差,因为用户会同时在训练集和验证集中,但通过时间分割,就可以按照现实世界的情况对模型进行检验,因为一旦模型被开发和部署,它就必须以一种主动学习范式进行更新或在过度衰退之前重新训练。

第7章 机器学习实用案例解析

7.1 数据分析^①

7.1.1 分析与验证

在数据处理方面,一个屡试不爽的方法就是:把任务清晰地分解成分析和验证两步。把数据处理分为分析和验证由著名的 John Tukey 首先提出,他强调要为实际数据分析设计简单的工具。在 John Tukey 看来,数据分析这一步要做的就是用摘要表(summary table)和基本可视化方法从数据中寻找隐含的模式。本章揭示如何用 R 中的基本方法来建立数据的摘要表,然后再讲解怎样从这个表中看出门道。之后,将列举 R 中一些用于可视化数据的工具和方法,同时,还会简要介绍基本可视化模式。

在开始第一次数据集分析之旅前,我们得提醒,一个真实存在的危险在时时窥伺你:你找到的模式也许并不存在。人类的大脑天生就能发现宇宙中的模式,甚至在不太可能出现模式的地方也能发现。看一眼白云,眨眼间就能发现云的形状,这跟知道多少统计学知识无关。很多人都深信自己可以在普通的字里行间,比如莎翁的戏剧,发现隐藏的信息。因为人类很容易发现一些经不起仔细推敲的模式,所以就不能只有数据分析这一步,必须还要有验证过程。可以这样来看待数据验证:数据分析阶段的可视化结果杂乱——有时甚至是毫无章法,这种情况让我们处理起来有些吃力,需要厘清思路,此时数据验证就会发挥作用。

数据验证通常有两种方法。

(1)如果你认为自己在一个新的数据集上发现了模式,那就用另一批数据来测试这个模式的正规模型。

(2)利用概率论来测试你在原始数据集中发现的模式是否只是巧合。

相比数据分析,因为数据验证对数学水平要求较高,所以本章只涉及数据

^①DREW CONWAY. 机器学习:实用案例解析[M]. 北京:机械工业出版社,2013.

的分析方法。也就是说在具体的案例中我们只关注数据的数值摘要(numeric-summary)和一些标准的可视化方法。我们讲到的数值摘要就是一些基本的统计项目:均值(mean)和众数(mode)、百分位数(percentile)和中位数(median)、标准差(standard deviation)和方差(variance)。我们要用到的可视化工具也是最基本的,你可能在统计学入门课程上已经学过了:直方图、核密度估计以及散点图。这些简单的可视化方法可能一度得不到重视,但是我们会让你相信,仅用这些基本的方法也能从数据中挖掘出有用的信息。本书后面的章节才会涉及许多较高深的技术,但是,要训练数据分析的直觉,最好就是用简单的方法去操作数据。

7.1.2 什么是数据

在介绍数据分析的基本方法之前,我们需要对“数据”这个概念统一认识。对“数据”这个概念所有可能的定义可以用一整本书来论述,因为对任何所谓的“数据集”你都可能有一堆重要的问题要问。例如,你可能经常想知道你手中的数据是如何产生的,你也想知道这些数据能否代表真正要研究的数据总体。通过研究亚马孙印第安人婚史,你可能已经掌握了关于他们社会结构的很多知识,但是能否从中得到一些适用于其他文明的知识却不得而知。对数据进行解释需要对数据来源有一定的了解。通常,唯一能区别因果关系和相关关系的方法就是要知道数据由何而来:是从实验中得到的,还是因没有实验数据而直接观察记录而来的。

虽然这都是些有趣的问题,而且我们也希望你终有一天能够具备这些知识,但是在本书中我们完全不会涉及数据采集。本章先假定数据分析这个微妙的哲学问题与预测问题完全无关,后者我们会用机器学习技术来解决。出于实用性考虑,我们要在本书余下内容中统一采用以下定义:“数据集”无非是一张充满数字和字符串的大表,表中每一行是现实世界中的单个观测记录,并且每一列是观测记录的一个属性。如果你很熟悉数据库,那么我们对数据的这个定义在直觉上应该符合你所熟悉的数据库表结构。如果你还担心你的数据集可能还不仅是单张表,那么我们先假定你已经用过 R 中的 merge、SQL 中的 JOIN 系列操作或者我们此前提到的其他工具,创建了类似单张表的数据集。

我们称这个定义为“数据方块”(dataasrectangle)模型。虽然这个观点大大简化了,但是却可以在数据分析时非常形象地帮助我们想出许多好主意,我们希望它可以让我们许多原本抽象的概念变得具体一些。“数据方块”模型还有另一个作用:它让我们可以自由地徜徉于数据库设计的思维和纯数学思维之间。

由于数据由矩形组成,因此我们可以轻松地把运算操作通过绘图的方式表示出来。一份数据的数值摘要就是把表中的所有行精简为只有几个数字——数据集的每一列常常就精简为一个数字。图 7-1 是这类型数值摘要的例子。

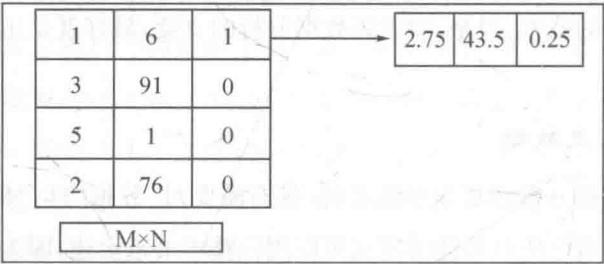


图 7-1 将每一列摘要成一个数字

与数值摘要相对的是另一种可视化摘要方法,它把数据集中所有数据的某一列概括成一张图。单列可视化摘要的例子如图 7-2 所示。

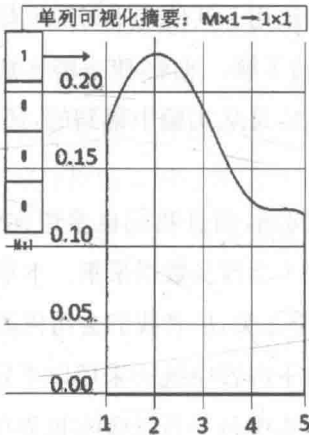


图 7-2 将一列摘要成一张图

除了分析单列的方法之外,还有很多方法用于分析数据集中多列之间的关系。例如,计算两列的相关性,可以把表中任意两列转换成一个数字,这个数字表征了这两列之间关系的强度,如图 7-3 所示。

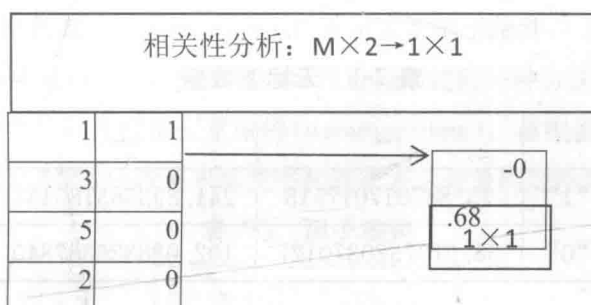


图 7-3 将一行摘要成一个数字

还有其他更为深入的方法。如果你觉得数据中存在很多冗余,那么你可能还需要减少其中的列数。图 2-4 列举了一个例子说明降维要达到的目的。

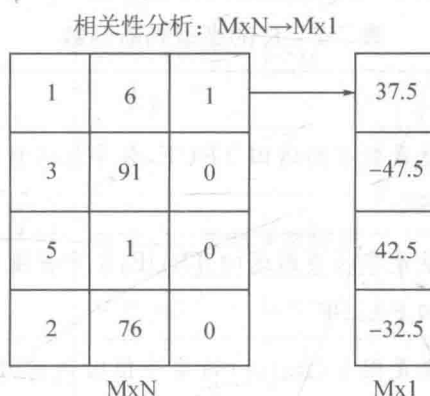


图 7-4 降维:将多列摘要成一列

如图 7-1~图 7-4 这四个图所示,摘要统计和降维是截然不同的两个方向:摘要统计要传达的是所有数据在某一列(某个属性)上的特点如何;降维要做的是把数据集中所有列(属性)转换成少数几列,得到的列数据对每一行来说都是唯一的。分析数据时,这两种方法都可能有用,因为它们都可以将海量数据变得一目了然。

7.1.3 数据推断的类型

在你要对新数据集进行下一步操作之前,首先要弄清楚这个表中的每一列所代表的含义。有人喜欢把这称作数据字典,就是说你可以在这里给数据集中每一列数据都存放一条简洁易懂的描述信息。例如,表 7-1 所示的是一个没有

标签的数据集。

表 7-1 无标签数据

...	...	...
"1"	73.847017017515	241.893563180437
"0"	58.9107320370127	102.088326367840

在没有任何提示信息的情况下,真的很难知道表中的数字表示什么含义。事实上,首先要搞清楚的是每一列的数据类型:第一列似乎仅包含字符 0 或 1,但它真的是字符串吗?当我们面对一份没有标签的数据集时,我们可能会用到 R 中的一些类型判断函数。三个最重要的类型判断函数如表 7-2 所示。

表 7-2 R 的类型判断函数

R 函数	简介
is.numeric	输入向量是数字则返回 TRUE,数字包括整数和浮点数;否则返回 FALSE
is.character	输入向量是字符串则返回 TRUE,R 中并没有单字符数据类型;否则返回 FALSE
is.factor	输入向量是因子(factor)的某个值时返回 TRUE,因子在 R 中用于表不分类信息;如果你用过 SQL 中的枚举类型,那么可以把因子看成类似的类型。它在内部隐藏的表示方式和语义上都与字符串向量有区别:R 中大多数统计函数的操作对象都是数值向量或因子向量,而不是字符串向量。输入不是因子的某个值时返回 FALSE

知道每一列的基本数据类型可能对于我们进行下一步操作非常重要,因为单个 R 函数常常因为输入数据的类型不同而进行不同的操作。在使用某些 R 内置函数之前,当前数据集中存储的这些 0 和 1 字符需要转换成数字,但是当使用另外一些 R 内置函数时,它们又会转换成因子类型。从某种程度上来说,这种在不同数据类型之间来回转换的做法是机器学习中处理分类时司空见惯的做法。许多真正充当标签或起分类作用的变量都以数学方式编码成数字 0

和 1。可以把这些数字想象成布尔值:0 表示正常电子邮件,1 表示垃圾电子邮件。这是用 0 和 1 对一个对象的定性属性进行描述的一种方法,这种方法在机器学习和统计学中叫作虚拟变量编码(dummy coding)。虚拟编码系统和 R 中的因子还是有区别的,后者是采用文字标签来表达对象的定性属性。

表 7-3 因子编码

MessageID	IsSpam
1	“yes”
2	“no”

表 7-4 虚拟变量编码

MessageID	IsSpam
1	1
2	0

表 7-5 物理学家编码

MessageID	IsSpam
1	1
2	-1

表 7-3~7-5 所示的是同样的数据,但是采用了三种不同的编码方案。

在表 7-3 中,IsSpam 就直接当作 R 中的因子处理,这样是一种表示定性区别的方法。而在实际中它既有可能以因子类型载入,也有可能以字符串类型载入,这取决于你所用的载入函数。每拿到一份新数据时,你都要先决定怎么用 R 处理每一列,再决定是以因子类型还是以字符串类型加载其值。

在表 7-1~7-4 中,IsSpam 仍然是一个定性概念,但是却以数字形式表示了布尔型的区别:1 表示 IsSpam 是 true,而 0 表示 IsSpam 是 false。实际上,很多机器学习算法都要求定性数据是这种编码方式。

最后,表 7-1~7-5 展示了对相同定性概念进行数值编码的另一种方式。在这种编码系统中,人们用+1 和-1,而不用 1 和 0。这种对定性概念编码的

风格很受物理学家青睐。当你看过的机器学习书籍够多,最终会遇到这种风格。但是本书会完全摒弃这种风格,因为在变量的不同表示方式间来回切换会导致无谓的混淆。

7.1.4 推断数据的含义

尽管你已经搞清楚了每一列的数据类型,但是对其含义可能仍然不甚了解。确定一张无标签的数字表到底在描述什么比大家想象的要困难得多。我们看看之前提到的表,见表 7-1。

假如我们告诉你以下这些信息:①每一行代表一个人;②第一列是一个虚拟变量,表示这个人是男性(值为 1)还是女性(值为 0);③第二列是每个人的身高,以英寸为单位;④第三列是每个人的体重,以磅为单位。知道这些之后,你再来看这张表,是否觉得豁然开朗?把这些数字放在适当的上下文之中,它们瞬间就变得有意义了,而且这也让你对这些数字所描述的对象有了一定的想法。

但是,遗憾的是,有时候你得不到这些解释性信息。遇到这种情况,我们能告诉你的方法就只有一个:用人类的直觉,再辅以大量的 Google 搜索。不过,在查看了这类数值摘要表或可视化摘要图之后,这两种摘要表中每一列的含义不明确,你的直觉会大幅改善,这也算是因祸得福。

7.1.5 数值摘要表

要弄清楚一份新数据的意义,最好的方法之一就是计算所有列的数值摘要。R 很适合完成这个操作。如果数据中只有一个列向量,summary 函数会产生出一些值,这些值的意义再明白不过了,你应该首先看一看:

```
data.file<-file.path('data','01_heights_weights_genders.csv')
heights.weights<-read.csv(data.file,header=TRUE,sep=',')
heights<-with(heights.weights,Height)summary(heights)
# Min. 1stQu. MedianMean3rdQu. Max.
# 54.2663. 5166. 3266. 3769. 1779.00
```

对一个数值向量调用 R 中的 summary 函数之后就会得到上述例子中这些数值结果:

- (1) 向量中的最小值;
- (2) 第一个四分位数(也称作第 25 个百分位数,即大于 25% 的数据中最小的那个);
- (3) 中位数(也称作第 50 个百分位数);
- (4) 均值;
- (5) 第 3 个四分位数(也称作第 75 个百分位数);
- (6) 最大值。

如果你想从数据中快速地得到一份数值摘要,这些值基本上就够了。还缺少的值是标准差,本章稍后会定义一份数值摘要表。在接下来的内容里,我们会教大家怎样单独计算 `summary` 函数所产生的每一个数值,并告诉大家它们都有什么意义。

7.1.6 均值、中位数、众数

区分均值和中位数是所有统计学入门课程中最乏味的内容之一。的确需要一些时间才能更加熟悉这些概念,但是当你真正处理数据时,我们相信你还是需要将两者区分开来。考虑到更好地让读者学到知识,我们试图通过两种不同的方式深化并强调这两个术语的意义。第一,如何通过算法方式计算均值和中位数。对大多数黑客来说,用代码表达想法比用数学符号更加自然,所以我们认为,自己写函数来计算均值和中位数,这样可能会让你更加透彻地理解这两个统计学概念的定义公式。第二,在本章末尾还会通过数据的直方图和密度图来揭示两者的区别。

计算均值相当简单。在 R 中,一般用 `mean` 函数计算均值。当然,把均值放在一个黑盒子函数里面计算不太能直观地体会到均值的含义,因此我们自己实现 `mean` 函数,命名为 `my.mean`。因为相关概念在 R 中已有其他函数可以计算:`sum` 和 `length`,所以仅需一行 R 代码。

```
my.mean<-function(x){return(sum(x)/length(x))
}
```

就这一行代码,均值的含义就已明了:只需把向量中的值求和,再除以长度。想必读者已经知道,这个函数用于产生向量 `X` 中所有数的平均数。均值

太容易计算了,因为它和列表中数字的排序毫无关系。

中位数就刚好相反:它完全依赖于列表元素的排序。在 R 中,一般用 `median` 函数计算中位数,但是我们要实现我们自己的版本,称之为 `my.median`:

```
my.median <- function(x) { sorted.x <- sort(x)
  if(length(x) %% 2 == 0)
  {
    indices <- c(length(x)/2, length(x)/2 + 1) return(mean(sorted.x[indices]))
  }
  else
  {
    index <- ceiling(length(x)/2) return(sorted.x[index])
  }
}
```

只看代码行数就知道计算中位数比计算均值要复杂一些。第一步,对向量排序,因为中位数本质上是有序向量中间的那个数。这也是中位数称为第 50 个百分位数、第 2 个四分位数的原因。第二步,一旦把向量排好序,就可以顺理成章地计算任何一个百分位数或四分位数,只需根据向量长度从某处将列表拆分成两段即可。要计算第 25 个百分位数(即第一个四分位数),只需把列表按照长度拆分并取前四分之一即可。

以长度为依据的不规范定义存在一个问题,那就是当数据长度是偶数时此定义就行不通。如果没有一个数明显地处于数据集中间,你需要为此专门设计一个计算方法。在上述例子的代码中,我们专门对这种长度为偶数的情况进行了处理:若列表长度为偶数,则在数据集中间有两个数——这两个数所在位置相当于列表长度为奇数的情况下的中位数——对这两个数求均值即是中位数。

为了解释得更为清楚,下面举一些简单的例子,第一个例子的中位数是中间两个数的均值,另一个例子的中位数就是中间那个数:

```
my.vector <- c(0, 100)
my.vector
# [1] 0 100
mean(my.vector)
# [1] 50
median(my.vector)
```

```
# [1] 50
my.vector <- c(0, 0, 100) mean(my.vector)
# [1] 33.33333; median(my.vector)
# [1] 0"
```

回到原来的身高和体重数据集, 计算身高的均值和中位数。顺便检验一下代码是否正确:

```
my.mean(heights)
# [1] 66.36756 my.median(heights)
# [1] 66.31807
mean(heights) - my.mean(heights)
# [1] 0
median(heights) - my.median(heights)
# [1] 0
```

在这个案例中, 均值和中位数非常接近。我们稍后会简单解释为什么这份数据的均值和中位数本就应该接近的。

介绍了以上两个重要的统计数值, 你可能奇怪为什么还不介绍众数。其中一个原因是: 众数不像均值和中位数那样总是可以用我们处理过的那些向量来简单定义。因为它不易自动化计算, 所以 R 中并没有计算数值向量众数的内置函数。

注意: 对任意向量定义众数是比较困难的, 因为若要用数值来定义众数就要求向量中的数字重复出现。但是当向量中的数值是任意浮点数时, 不太可能任意值都会在向量中重复出现。因此, 众数在许多种数据集上仅有可视化定义。

总之, 如果你仍不确定众数的数学含义或其理论意义, 你就假定它是在数据集中出现次数最多的那个数。

7.1.7 分位数

前面已经说过, 中位数就是出现在数据集的 50% 那个位置的数。为了对数据范围有更深入的认识, 你可能想知道数据中最小的那个数是什么。这就是

数据集的最小值,这可以用 min 函数计算:

```
min(heights)
```

```
# [1] 54.26313
```

而数据集中最高/最大的那个数则可以用 max 计算:

```
max(heights)
```

```
# [1] 78.99874
```

两者联合确定了数据的范围:

```
c(min(heights), max(heights))
```

```
# [1] 54.2631378. ; 99874 range(heights)
```

```
# [1] 54.2631378. 99874
```

对此可以从另一个角度进行理解:在数据集里,min(最小值)指的是 0% 的数据都小于它的数字,max(最大值)指的是 100% 的数据都比它小的数字。由此可以自然地联想到:如何找出数据集中 $N\%$ 的都小于它的数? 答案是使用 R 中的 quantile(分位数)函数。第 N 个分位数就表示数据集中有数据小于它。

默认情况下,quantile 会告诉你数据集的 0%、25%、50%、75% 以及 100% 位置处的数据:

```
quantile(heights)
```

```
# 0% 25% 50% 75% 100%
```

```
# 54.2631363. 5056266. 3180769. 1742678. 99874
```

要得到其他位置的分位数,需要给 quantile 的另一个参数 probs 传入截取位置:

```
quantile(heights, probs=seq(0, 1, by=0.2))
```

```
# 0% 20% 40% 60% 80% 100%
```

```
# 54.2631362. 8590165. 1942267. 4353769. 8116278. 99874
```

这里用 seq 函数在 0~1 之间产生了一个步长为 0.2 的序列:

```
seq(0, 1, by=0.2)
```

```
# [1] 0.00. 20. 40. 60. 81. 0
```

分位数虽然在统计学传统教程中不如均值和中位数那样受重视,但其作用不容忽视。如果你在运营一个客服部,并记录用户反馈的响应时间,那么可能

你关心前 99% 顾客情况的收益远大于只关心中位数个数顾客情况。如果数据形状比较奇怪,只关心平均数个数的顾客情况就更不能反映整体情况了。

7.1.8 标准差和方差

从某种意义上说,均值与中位数都是一组数的“中间的数”:中位数,顾名思义,就是一组数据的中心位置,而均值实际上则是列表中所有数值加权之后的中心。

不过,对于一份数据而言,它的集中趋势可能只是你关心的一个方面。了解通常数据与期望值有多大偏移也同样重要,这称为数据的散布。定义数据的范围有很多方式,比如前面提到的 range 函数:数据范围由 min(最小值)和 max(最大值)决定。但是要合理地定义散布程度,还需要以下两个因素。

(1) 散布程度覆盖的应该只是大多数数据,而非全部数据。

(2) 散布程度不应该完全由数据的两个极端值决定,这两个极值通常只是异常值而无法代表数据集整体情况。

Min(最小值)和 max(最大值)通常都是异常值,因此用它们来定义散布程度相当不稳定。换个角度思考,假设我们保持这两个极值不变而改变其余数据,那会怎样? 实际上你可以随意地去除其余数据而仍然保持最大值和最小值不变。也就是说,不论你是 200 万个还是两个数据点,建立在最大值和最小值基础上的定义都只依赖于数据集的两个数据点。因为我们不能相信任何对大部分数据点都不敏感的摘要,所以要用更好的方式来定义数据集散布程度。

对数据集进行数值摘要已有很多可行的方法。例如,可以计算包含 50% 数据的范围是什么,围绕中位数的数是什么。用 R 很容易统计出这些信息:

```
c(quantile(heights, probs=0.25), quantile(heights, probs=0.75))
```

或者可把范围扩大,找一个覆盖 95% 数据的范围:

```
c(quantile(heights, probs=0.025), quantile(heights, probs=0.975))
```

这些的确可以很好地衡量数据的散布程度。当你用到更高深的统计学方法时,此类范围定义方法比比皆是。但是历史上的统计学家们却用过一个不太一样的散布程度衡量方法:该方法明确地定义为方差(variance)。大致来说,这个定义是为了衡量数据集里面任意数值与均值的平均偏离程度。若是一名黑

客, 他们要写一个自己的方差计算函数, 而不是写方差的数学计算公式:

```
my. var<-function(x){m<-mean(x)
return(sum((x-m)A2)/length(x))
}
```

和之前一样, 把这个函数与 R 的 var 函数作比较, 以确保代码正确:

```
my. var(heights)-var(heights)
```

实现的函数和 R 内置函数 var 的执行结果有偏差。从理论上, 可以找一些理由来解释: 即使不考虑浮点运算的精度问题, 仍然还会有偏差。事实上, 另一个重要原因是函数的实现方式与 R 内置函数所用的方法不同: 在正规的方差定义中, 除数并不是向量的长度, 而是向量的长度减 1。之所以这样做, 是因为从经验数据估算的方差会由于一些细微原因比其真值要略小。要修补这个误差, 假设数据集有 n 个数据点, 你会自然地想到用比例因子 $n/(n-1)$ 与方差估计值相乘, 由此更新后的 my. var 函数实现:

```
my. var<-function(x){m<-mean(x)
return(sum((x-m)A2)/(length(x)-1))
}
my. var(heights)-var(heights)
```

用这个版本的 my. var 函数计算方差, 可以和 R 内置方差估算函数的结果完全一致。上文关于浮点运算有偏差的观点在我们进行长向量运算时很常见, 但是本例中的向量长度还不涉及此问题。

用方差衡量数据集散布程度合乎情理, 但是它的值几乎比数据集中任何一个值都要大很多。用一个很直接的方法就可以看到这个现象, 看看与均值相差正负单位方差之间的数值:

```
c(mean(heights)-var(heights), mean(heights)+var(heights))
#[1]51.5640981.17103
这个范围实际上比源数据集的范围要大:
c(mean(heights)-var(heights), mean(heights)+var(heights))
#[1]51.5640981.17103range(heights)
#[1]54.2631378.99874
```

之所以目前的数据范围超出原始数据范围,是因为定义方差的方式是衡量列表中数据与均值的平方距离,而没有对其进行开方。为使一切都回归原始的范围,需要用标准差(standard deviation)代替方差,即方差的平方根:

```
my.sd<-function(x){return(sqrt(my.var(x)))
}
```

在进行下一步操作之前,最好检验一下你的实现和 R 中的内置函数是否一致,此函数在 R 中叫作 sd:

```
my.sd(heights)-sd(heights)
```

因为现在计算的值在正确的范围内,所以有必要重新估算数据范围,看看偏离均值单位标准差的数值:

```
c(mean(heights)-sd(heights),mean(heights)+sd(heights))
#[1]62.5200370.21509range(heights)
#[1]54.2631378.99874
```

既然用的是单位标准差,而不是单位方差,那么得到的数据范围就轻松落入极差(range)范围内。要对数据内部集中程度有个基本认识,方法之一便是对比基于标准差的范围和基于分位数的范围:

```
c(mean(heights)-sd(heights),mean(heights)+sd(heights))
# [1]62.5200370.21509
c(quantile(heights,probs=0.25),quantile(heights,probs=0.75))
# 25% 75%
# 63.5056269.17426
```

用 quantile 函数可以看到,大致有 50% 的数据落在距离均值正负一个标准差之间的范围之内。这是个典型的数据特征,对这份身高数据来说更是如此。但是要最终精确地刻画数据的形状,还需要对数据进行可视化操作,并定义一些正式用语来描述数据的形状。

7.2 智能收件箱

7.2.1 次序未知时该如何排序

二分类的概念就是把对象判定为两个类别中的其中一个。在很多情况下,

能够做出这样的辨别,我们就满意了。但是,如果同一个类别中每个对象并非被同等创建,并且我们想在这个类别中对它们进行排序那又该当如何? 举个简单的例子,假如我们想说某一封邮件垃圾倾向最高,另一封的垃圾倾向程度次之,或者用其他有数的方式去对它们进行区分,那该怎么办呢? 想一下,我们不仅要过滤垃圾邮件,还要将更重要的邮件置顶在收件箱列表中。这个问题在机器学习中很常见。

通过产生一组规则对一个对象列表排序,这在机器学习中越来越常见,只不过你可能还未从专业角度思考过。更可能听说过类似推荐系统的东西,它就是在后台对产品进行了排序。即使你可能连推荐系统也没听说过,但是你肯定在某些场合使用过或者与之交互过。一些非常成功的电子商务网站已经从中尝到甜头了,他们利用其用户数据为用户推荐可能感兴趣的其他产品。

举个例子,如果你曾经在亚马逊(Amazon.com)上买过东西,那么你就与推荐系统交互过。亚马逊要解决的问题很简单:你最可能购买他们列表上的哪些库存商品? 这种说法的隐含意义就是:亚马逊列表上的库存商品对每个用户来说都有一个特定的顺序。同样,在Netflix.com上有海量的DVD可以出租给用户。为了让用户能在这个网站上找到尽可能多感兴趣的DVD,Netflix部署了一套复杂的推荐系统用于为人们提供租赁建议。

这两家公司的推荐系统都基于两种。第一种是库存商品本身的数据。对亚马逊来说,如果商品是一台电视机,那么这种数数据可能包含其他类型(比如:等离子、LED、LED)、制造商价格等。对Netflix来说,这个数据也许是电影的题材、演员阵容、导演、片长等。第二种是消费者浏览和购实行为数据。这类数据可以都助亚马逊了解大多数用户在购买一台新等离子电视机时需要什么配件,可以帮助Netflix了解George A. Romen的粉丝们最经常租赁的是哪些浪漫喜剧。这两种数据的特征很容易确定,我们知道什么样的数据是分类数据,比如产品类型或电影题材,而且用户产生的数据以及购买/租赁记录以及评分等形式已经被很好地结构化。

在进行排序时,通常已有明确的输出实例,这种机器学习问题一般称为有监督学习(supervised learning)。与之相对的是无监督学习(un-supervised learning),无监督学习在开始处理数据时预先并没有已知的输出实例。要对两

者的区别理解得更透彻,可以把有监督学习看成是经过指导的学习过程。比如,你要教一个人烤樱桃派,你可能会给他一份食谱,然后让他尝尝自己做出来的派。尝过之后,他可能会根据味道对配料再做一些调整。有了一份用过的配料表(也就是输入)。也有了做出来的味道(也就地输出),这意味着他可以分析每一种配料有多大的作用,从而找到一份制作美味樱桃派的完美食谱。

相反,如果你只知道炸豆泥可以搭配炸玉米粉田饼,而烤樱桃可以和面团搭配,那么你也也许能够把这些调料划分成不同的类型,有些可以用来做墨西哥菜,有些可以用来做美式甜点。实际上最常用的无监督学习就是聚类,聚类就是把对象按照其相同点或不同点分别归入一定数目的组内。

7.2.2 按优先级给邮件排序

哪些因素决定了邮件的重要性?要回答这个问题,先退一步想想什么是邮件。首先,它是一种基于事务的媒介。人们在不断地收发消息。因此,要确定邮件的重要性,需要关注事务本身。在垃圾分装中,可以利用所有邮件的静态信息来决定其类型,而与之不同的是,要根据重要性给邮件排序,必多重关注往来事务本身的动态信息。具体地讲,我们要做的就是确定一个人在收到一封新邮件后马上处理的可能性有多大。换句话说,假设已经选择好特征集,那么收件人会在短时分内处理这封邮件的可能性有多大。

这个问题涉及的一个新的关键维度就是时间。在个基于事务的情境中,要对事件的重要性排序就得考虑时间因素。用时间来决定邮件的重要性,很自然的方法就是计算收件人在收到邮件后过了多少时间才处理这封邮件。在给定特征集下,这个平均时间越短,那么这封邮件在所属类型的重要性就越高。

这个模型的隐含假设就是越重要的邮件越早被处理。直观上感觉这是讲得通的。每个人在盯着收件箱浏览邮件时,都可以将需要立即处理的和可以稍后处理的邮件分辨得清清楚楚。我们这样自然而然做出的区分是要让算法实现的功能。但是在开始之前,必须确定邮件内容里的哪些特征可以很好地表征邮件优先级。

7.2.3 文本分析——经典的垃圾邮件过滤系统

文本分析算法是一系列算法的合称,在文本分析中必须要完成的工作的有

分词、清洗和在文本中发现信息。这些工作可以使用 K 均值算法、支持向量机或朴素贝叶斯算法完成,本小节通过垃圾邮件过滤系统的构建展示了不同算法是如何用于文本分析工作的,以及文本分析工作中需要注意的地方和文本分析工作能够解决的常见问题。

(1) 大数据时代需要文本分析工作

大数据时代最显著的一个特点就是能够在海量的数据中获取有用的知识。数据分析师之所以能够利用海量数据,不仅是因为现代世界数据量的爆炸性增长,还因为我们开始利用以往忽略的非结构化数据。这些非结构化数据包括图像、音频、视频、文字等,它们占据了现有数据量中的绝大部分。

在大数据时代到来前,我们对于非结构化数据的利用是低效的。比如犯罪事件发生后,警察会调出视频记录查看案件发生时的情况,这时几百个小时的视频记录中只有几分钟被利用起来了,这种利用是非常低效的。但是现在我们可以用这些视频来训练人脸识别算法,并在新的监控视频中智能识别出罪犯,这种基于大数据和算法的识别就比较高效了。对图片、视频的分析工作统称为图像分析,同理,对文字的分析工作统称为文本分析。由于摄像机等监控设备往往由政府部门掌管,因此图像分析通常局限于政府公共事务领域,比如智能人脸识别系统,以及自动驾驶技术。而文本分析则不然,网络上存储的文本数据大部分都是可以随意被个人或企业获取的,如论坛上的用户发言、推特内容、微博内容、新闻网站发布的文章资讯等,因此文本分析能够解决的问题要丰富得多。

(2) 垃圾邮件过滤中的分词技术和词集模型

分词是文本分析的第一个步骤。在英文中,单词与单词之间被空格分隔开,因此英文文档的分词工作是较为简单的,只需在每个空格处切割文本即可。但在中文中,字与字、词与词之间没有明显的分割点,因此需要一个模型来完成这件事。

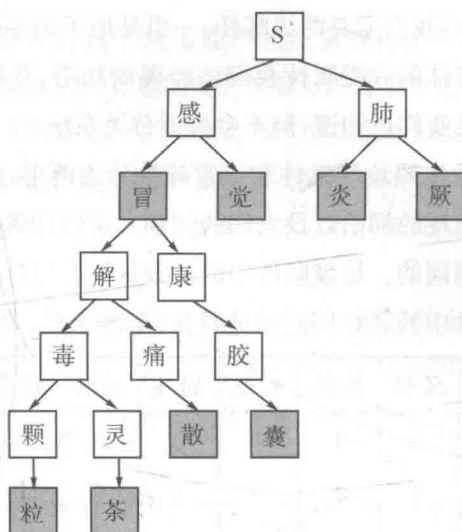


图 7-5 机械分词模型示例词典

图 7-5 是一个机械分词模型的示例词典，它一共收录了感冒、解毒颗粒、解毒灵茶、解痛散、康胶囊、感觉、肺炎、肺厥等八个词语，使用树状结构表示，当新的词语输入模型时，它就被用来和词典相匹配，从而完成分词工作。

以“感冒解毒颗粒”这一短语为例，模型首先提出短语的第一个字符“感”，并在词典中找到“感”字的位置，然后接着将短语后续的内容与词典中收录的词汇相匹配，由于“冒”字是粉色的终止字符，故“感冒解毒颗粒”中提出的第一个词汇就是“感冒”。然后原始短语中删除“感冒”，从“解”字重新开始匹配，最终完成分词过程。

在实际应用中，分词词典几乎包含了所有已知的常见词汇，因此中文句子中的所有词汇都可以切分出来。这种按照句子顺序从前向后切分的方法完全依赖词典的完备性，而不涉及任何概率知识，因此叫作机械分词模型。

机械分词模型只能按照从前向后的顺序切割，因此“我喜欢上海南了”这句话就会切割成“我、喜欢、上海、南、了”这种形式，为了避免这种错误，我们通常在机械分词模型的基础上引入神经网络或贝叶斯决策方法，通过使用概率来寻找正确的分词方法。即“喜欢、上、海南”这种组合出现的概率大于“喜欢、上海、南”，因此我们选择将句子分割为“我、喜欢、上、海南、了”。

解决了分词问题后，垃圾邮件分类系统的构建就已经完成一半工作了。在

垃圾邮件分类系统中,我们需要两组邮件,一组是用于训练的邮件,一组是用于测试的邮件。训练邮件的分类属性我们已经提前知道,并且会告诉分类系统。测试邮件的分类属性我们也知道,但不会告诉分类系统。

我们将训练邮件按照垃圾邮件和正常邮件分为两组,为它们分词后,统计出每组邮件中高频出现的词语以及它们的词频。可以理解,这两组邮件中的高频词汇是绝对不会相同的。垃圾邮件中的高频词汇会是尊敬的、顾客、活动、促销、欢迎等,正常邮件中的高频词汇则是朋友、宠物、约会、有趣等。

	尊敬的	顾客	活动	促销	欢迎	朋友	宠物	约会	有趣	属性
1	1	1	0	1	1	0	0	0	1	垃圾邮件
2	1	1	1	1	0	1	0	0	0	垃圾邮件
3	1	0	1	0	0	0	0	0	0	垃圾邮件
频数	3	2	2	2	1	1	0	0	1	
频率	0.25	0.17	0.17	0.17	0.08	0.08	0	0	0.08	
4	0	0	0	0	0	1	0	1	0	正常邮件
5	0	0	0	0	0	1	1	1	1	正常邮件
频数	0	0	0	0	0	2	1	2	1	
频率	0	0	0	0	0	0.33	0.17	0.33	0.17	

图 7-6 训练邮件的词频统计图

图 7-6 是根据五封训练邮件统计得到的词频统计图。其中 0 表示某个词汇在某封邮件中没有出现,1 则表示某个词汇在某封邮件中出现了。我们将垃圾邮件和正常邮件分开统计,最终得到每个词汇在每种邮件中出现的频率。当训练邮件个数较多时,我们将频率直接当作概率来使用。

根据图 7-6,我们认为尊敬的、顾客、活动、促销、欢迎、朋友、宠物、约会、有趣等词汇在垃圾邮件中出现的概率分别是 0.25、0.17、0.17、0.17、0.08、0.08、0、0、0.08,在正常邮件中出现的概率分别是 0、0、0、0、0、0.33、0.17、0.33、0.17 等。那么当新邮件中出现“尊敬的”这一词汇时,我们就认为它是垃圾邮件的概率是 0.25。当新邮件出现时,我们首先对其进行分词,并根据概率公式 $p = \sum_{i=1}^9$

$c_i p_i$, 我们可以计算新邮件属于垃圾邮件或正常邮件的概率, 比如对于一个含有尊敬的、欢迎、朋友、约会、有趣的邮件来说, 它属于垃圾邮件的概率为 $1 \times 0.25 + 0 \times 0.17 + 0 \times 0.17 + 0 \times 0.17 + 1 \times 0.08 + 1 \times 0.08 + 0 \times 0 + 1 \times 0 + 1 \times 0.08 = 0.41$, 属正常邮件的概率为: $1 \times 0 + 0 \times 0 + 0 \times 0 + 0 \times 0 + 1 \times 0 + 1 \times 0.33 + 0 \times 0.17 + 1 \times 0.33 + 1 \times 0.17 = 0.83$ 。由于 0.83 大于 0.4, 因此新邮件属于正常邮件。

我们根据训练邮件得出模型参数后, 还需在测试集上测试模型的准确率, 即使用模型为测试集分类, 并观察预测结果和实际结果的差异。这种交叉验证方法在机器学习中是不可或缺的一个步骤。

(3) 文本分析小结

本小节介绍的垃圾邮件过滤系统是一个经典的文本分析问题。为了解决这一问题, 本小节着重讲述了如何使用机械分词方法为中文分词, 以及如何将概率相加计算得到每一个新邮件最终属于垃圾邮件或正常邮件的可能性。在文本分析中, 还有许多值得补充的内容。比如对于分词来说, 专家系统分词模式、神经网络分词模式以及路径选择分词模式等都是机械分词模式的加强版, 它们在精确度方面要优于机械分词模式, 但机械分词模式的速度是最快的。无论是哪种分词模式, 分词结果的准确与否都极大地依赖于分词词典的完备性。而对于概率计算方法来说, 一方面我们使用的是词集模型, 即考虑某一单词在某一邮件中是否出现, 而没有使用词袋模型, 即考虑某一单词在某一邮件中出现的次数; 另一方面我们使用的是简单的概率相加公式, 它计算简洁, 但最终结果并不是落在 0 到 1 之间, 因此具有缺陷。作为简单概率公式的改进, 我们也可以使用概率相乘公式, 或朴素贝叶斯概率公式来计算新邮件落入不同邮件分类中的概率。

此外, 我们还可以选择其他解决问题的思路来处理垃圾邮件分类问题。比如将每一个特征词看作一个维度, 使用支持向量机或 K 均值聚类算法来完成分类。但由于在垃圾邮件分类问题中, 我们往往提取的特征数非常多, 且数据中存在很多 0, 因此支持向量机或 K 均值算法的效果并不会比朴素贝叶斯算法的效果好。垃圾邮件分类问题的重点在于合理的提取用于计算概率的特征词, 而对于其他文本分析问题来说, 使用数据表示文本信息的方法未必是唯一的, 在具体问题中如何将文字性的问题转化为用公式表达的问题才是重点。

总的来说, 大部分文本分析问题都是垃圾邮件分类系统的变形或复杂化。

7.3 搜狗输入法案例解析^①

搜索引擎是每个人都十分熟悉的一项服务,具体来说,首先用户在搜索栏中输入搜索关键词,如“春季碎花外套”,其次搜索引擎将关键词切割为“春季”、“碎花”和“外套”,再次搜索引擎从所有的网页中抓出网页标题包含这三个关键词网页,并按照一定的规则排列展示给用户。以上是传统搜索引擎的工作模式,这种工作模式合情合理,但仍具有疏漏。比如在上文的例子中它只会推荐标题是“春季碎花外套”的网页,而不会推荐标题是“春季小花外套”的网页,并且认为标题是“春季碎花上衣”的网页和标题是“春季碎花短裤”的网页同样不重要。

我们不能一条一条地告诉搜索引擎“碎花”等同于“小花”、“上衣”比“外套”更重要等网页排序规则,这是因为首先这些规则非常多,其次这些规则总是在变化。好在语义搜索引擎能够通过学习,智能化的分辨关键词之间的细微不同,并捕捉到用户搜索关键词的真正含义。从根本上解决以上问题。

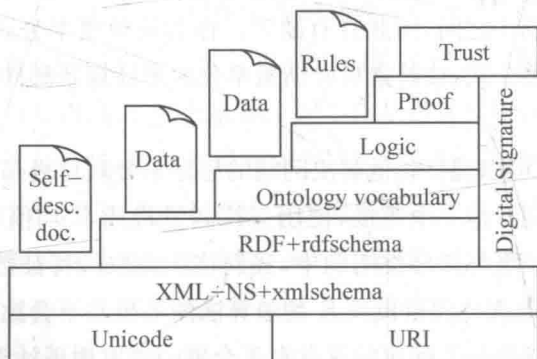


图 7-7 语义搜索引擎体系结构

① 张艳敏. 基于机器学习的行人检测[D]. 石家庄: 河北师范大学, 2012.

图7—7是一个语义搜索引擎的体系结构,它分成七层,越往下越接近底层技术,越往上越接近具体的搜索规则和网页。

Unicode 和 URI 所代表的字符集层以及 XML+NS+xmlschema 所代表的根标记语言层是搜索引擎的基础设施,RDF 层则提供词汇嵌入的框架,这一层将多种词汇集中起来描述 Web 资源。这三层都属于搜索引擎的基本构架,它们支撑着搜索引擎的正常运转。从 Ontology Vocabulary 层向上就是语义搜索引擎的核心内容了。Ontology Vocabulary 的意思是本体词汇表,这一层用于支持知识库的搭建。本体是一种术语的集合,它把现实世界的物体用一组一组的术语描述出来。例如,上衣这一本体可以由季节性、长度、薄厚、风格、颜色等概念构成,而季节性这一概念又成为一个单独的本体,由春、夏、秋、冬构成,长度由短、中、长构成……将这些本体合起来,就得到了本体词汇表。

本体词汇表可以由半监督的形式形成,即根据以往累积的用户搜索记录提取得到高频率出现的词汇,并人工汇总本体词汇表。不同语义搜索引擎的本体词汇表很可能不同,如沃尔玛百货搜索引擎中就不会收录关于“数据分析”等科技类的本体词汇。有了本体词汇表,知识库就能够将某一确定的关键词转化为一组概念的集合,并搜索到相关联的网页。

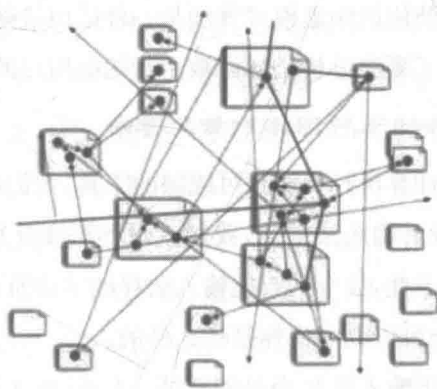


图 7-8 网页文档和网页数据的对应关系

用户在搜索栏输入关键词后,语义搜索引擎将找出与关键词相似度高的词语和相关度高的词语,相似度高的指的是两组词语具有互相代替的关系,比如“碎花”可以用“小花”代替;相关度高指的是两组词语总是伴随出现,比如“数据分

析”和“数据挖掘”总是同时出现。语义搜索引擎将网页中的相似词和相关词找出来,并根据这些词语的多寡按照一定规则排序。

图 7-8 展示了搜索引擎怎样从网页文档中提取出有效数据来。图中每个纸张图形都代表一个真实的网页文档,网页文档中包含的相关词和相似词则用图形中的蓝色圆点表示。网页文档中圆点越多,该文档的重要性就越高。向搜索引擎中引入相似度和相关度的概念将改变原有网页的排列顺序,有些网页未必包含了所有的用户搜索关键词,但只需包含大量相似关键词和相关关键词,即可获得较好的排名。

图 7-7 中的 Logic、Proof、Trust 层用于规则的推理和验证。它们通过逻辑推理对关键词、关键词之间的关系以及搜索结果排列顺序的合理性进行验证。这种验证技术主要基于 Proof 交换和数字签名技术,它一方面将钓鱼网站和伪装成优质网站的劣质网站剔除;另一方面预防不合理的网页排序结果,从而确保了良好的用户体验。

7.3.2 顺序分析——搜狗输入法的智能纠错系统

顺序分析是关联分析的一种,它能够在大量数据集中发现数据的关联性或相关性。顺序分析关心的数据的纵向排列,即一件事情发生后紧接着会发生什么事情。顺序分析所使用的频繁模式算法是一种适用且简单的算法,本小节以搜狗输入法为例讲解了顺序分析是如何被用于挖掘用户的固有输入习惯的。

7.3.3 搜狗输入法的王牌词库和智能算法

输入法是人类和计算机打交道不可或缺的工具,我们的电子设备上什么软件都可以没有,但是没有输入法的话,我们就没办法和计算机愉快的沟通。但输入法的功能并不止于此,如今的智能输入法存储了你所有的按键数据。不夸张地说,这些数据完全可以刻画出你是什么样的人。

输入法的优劣在于输入法是否足够智能。如今市面上流行的输入法有百度输入法、腾讯输入法、微软输入法和搜狗输入法等。平心而论,尽管搜狗公司的浏览器做得实在是不怎么样,但搜狗公司的输入法确实是最好的,而且比其他输入法好了不止一点点。截至 2015 年 5 月,搜狗输入法已经更新到了 7.5 版本,现如今,搜狗输入法支持的经典模块有简拼输入、智能纠错和热门词汇。

其他输入法固然也支持这些功能,但它们都不如搜狗输入法做得好。

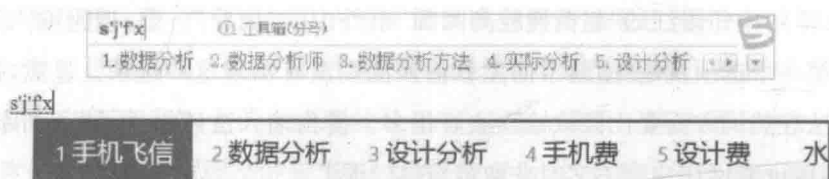


图 7-9 搜狗输入法和微软输入法简拼输入对比图

图 7-9 是搜狗输入法和微软输入法在简拼输入方面的对比图首字母“sjfx”作为测试对象紧接着列出了“数据分析师”词条,搜狗输入法通过分析以往的输入记录,而微软输入法的第一个选择是中规中矩,但并不够人性化。当然记录可能是导致微软输入法表现不好的原因之一。

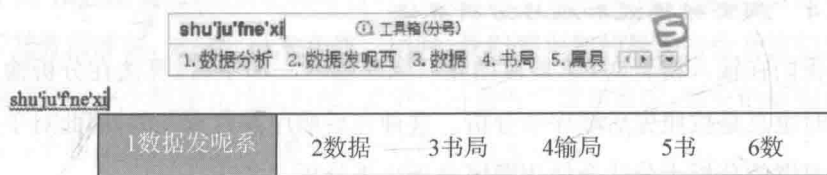


图 7-10 搜狗输入法和微软输入法智能纠错对比图

图 7-10 是搜狗输入法和微软输入法在智能纠错方面的对比图。测试输入均为“shujufnexi”其中 n 和 e 的位置打错了,正确输入应该是“shujufenxi”。搜狗输入法辨别出了这种错误,给出了“数据分析”这一选项,微软输入法则没有。

智能纠错模块不仅支持对按键顺序混乱的纠正,也支持对按键缺漏和拼音错误的纠正。比如“shujufexi”这一输入中少了一个 n,但输入法仍能判别出这是“数据分析”的输入;而输入“shenmo”后,搜狗输入会给出“什么(shenme)”的选项,在纠错的同时向用户支出错误。这些功能的实现一方面和用户个人的输入习惯相关,另一方面也和搜狗输入法中累计的上千用户的输入记录相关。

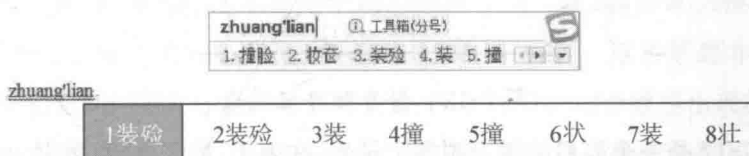


图 7-11 搜狗输入法和微软输入法热门词汇对比图

图 7-11 是搜狗输入法和微软输入法在热门词汇方面的对比图。测试词汇是 2015 年中旬爆红的“赵薇撞脸高圆圆”事件中的“撞脸”一事,搜狗输入法给出的第一个选项就是“撞脸”,而微软输入法则没有出现这一选项。显然,搜狗输入法在热词方面要比微软输入法好很多。搜狗输入法还提供了“云词库”功能,以保证联网用户能够实时获取最新热门词汇。

除了以上功能外,搜狗输入法也提供实时翻译、细胞词库、字符表情等多项功能。所有这些眼花缭乱的功能都依赖于搜狗输入算法的正常工作,而算法的正常工作又依赖于搜狗的庞大词库。这种简拼输入和智能纠错的功能尽管看起来并不起眼,但其实里边蕴含的算法原理是十分深刻的,本小节所关心的正是搜狗输入法中的顺序分析算法。

7.3.4 频繁树模式和顺序分析算法

我们在输入拼音时,总是遵循先后顺序输入一串字符,算法在分析输入的字符时也总是按照先后顺序来分析。这种先后顺序是有意义的,因此对于用户打字习惯的分析十分适合使用顺序分析法来分析。

搜狗输入法累积了用户的大量数据,这是它高度智能化的基础。在进行顺序分析前,搜狗输入法首先需要分解用户输入的序列,比如将 sjfx 分解为 s'j'f'x,将 shujufx 分解为 shu'ju'f'x。这种分解依赖于已有的词库,即工程师需要首先告诉输入法 shu 是一个字符,而 sj 就是两个单独的字符。

序号	顺序
1	s,j,f,x
2	s,j,f,x,f,f
3	s,j,w,j
4	s,h,s
5	s,j,f,x,s

图 7-12 拼音顺序记录表

图 7-12 是一个简易的拼音顺序记录表,在表中,数据分析、数据分析方法、数据挖掘、上海市、数据分析师的简拼顺序分别出现了一遍。顺序分析的任务

就是在这些记录中发现频繁出现的那些顺序,并在新顺序出现时,将它与已知频繁顺序相比较,找出新顺序的潜在规律或异常。在顺序分析中,支持度和置信度是最重要的两个概念。支持度指的是顺序出现的频率。比如单个字 s 在集合中出现了 5 次,它的支持度就是 5,而 sj 这一顺序一共出现了 4 次,它的支持度就是 4;置信度则指的是当某一顺序出现后,其他顺序紧接着出现的频率。比如 sj 顺序出现后,紧接着出现 fx 的置信度是 3,紧接着出现 $fxff$ 的置信度是 1。将支持度除以集合的总数或置信度除以该前缀所对应的子集总数后,这两个值就落入了 0 到 1 之间,比如 sj 的支持度 $4/5$, $sjfx$ 的支持度是 $3/5$,而 $sjfx$ 相对 sj 的置信度则是 $3/4$ 。此时的支持度和置信度可以直接衡量某一顺序出现的概率,而不需再结合集合总数进行分析。值得指出的是,支持度和置信度是两个单独的概念,二者之间没有任何关联,一个顺序的支持度高,置信度未必高,其置信度高,支持度也未必高。同时,我们感兴趣的是支持度和置信度同样高的顺序,一个顺序支持度低时,说明它出现的次数非常少,对它的研究没有意义;一个顺序置信度低时,说明关于它的结论不可靠。

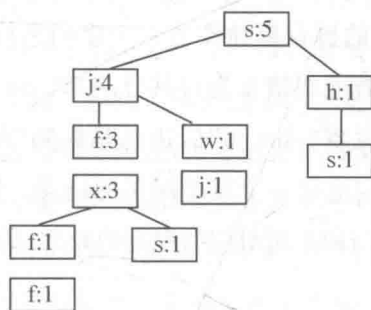


图 7-13 拼音顺序记录的频繁树

显而易见,顺序分析的本质思想是使用概率的知识来解决问题,一个顺序出现的概率大,则新顺序符合这个顺序的概率就大。比如我老是要输入“数据分析”这个词组,在我的记录里这个词组的频率就非常高,因此当我再次输入“sjfx”时,搜狗输入法就知道我想要输的是“数据分析”。这种思想非常容易理解,困难的是如何把顺序合理地转化为数字,从而统计每个顺序出现的概率呢?图 7-11 是拼音顺序记录的频繁树表示,频繁树模式是一种非常好用的关联分析方法。

在图 7-13 中, 每一个节点都由一对字母和数字组成, 字母表示该节点的输入字符, 数字表示该节点的支持度。这棵树从上往下读, 最上边的节点是根节点, 越靠近根节点, 支持度就越大。根据图 7-13 将所有顺序按照支持度从大到小排列后有 $\{s:5\}$ 、 $\{sj:4\}$ 、 $\{sjfx:3\}$ 、 $\{sjfxff:1\}$ 、 $\{sjfxs:1\}$ 、 $\{sjfxwj:1\}$ 、 $\{shs:1\}$ 等七个顺序。在顺序分析中, 我们通常不会保留所有的顺序, 而只是保留那些支持度高的、有意义的顺序。比如在本例中, 我们将阈值设为 1, 即只保留顺序 $\{s\}$ 、 $\{sj\}$ 以及 $\{sjfx\}$ 。同理, 我们计算每个顺序的置信度, 并设一个阈值来选出置信度较高的顺序。当用户输入 s 后, 根据频繁树, 我们将认为用户下一个将要输入的字符为 j ; 当用户输入 sj 后, 根据频繁树, 我们将认为用户即将输入 fx 。简拼输入是基础的顺序分析, 它所涉及的顺序元素全都是单独的字符。在智能纠错中, 顺序分析所涉及的元素则夹杂了多个字符, 比如 $shujufx$ 这一序列应当拆分为 shu 、 ju 、 f 、 x 这四个元素。将一个汉字的拼音视为一个整体能够简化计算, 并且, 这也是符合认知习惯的。

智能纠错寻找元素排列规律的方法与简拼输入并无不同, 它寻找拼音错误的方法在于将每个汉字的拼音单独作为一个序列进行分析。比如对于“ $shujufnxi$ ”这一序列来说, 智能纠错首先将其分割为“ shu 、 ju 、 f 、 ne 、 xi ”这一序列, 并搜索常见排列顺序, 发现“ shu 、 ju ”后边总是跟随“ fen 、 xi ”, 并比较“ fne ”和“ fen ”的相似度, 最终确定这是一个需要纠正的错误。热门词汇功能则在于调高了顺序分析算法中热门词汇所对应的序列的频率, 从而保证热门词汇能够排在更靠前的位置。

7.4 基于机器学习的行人检测^①

7.4.1 基于机器学习的行人检测综述

近年来, 通过不懈的研究, 在行人检测领域涌现了大量优秀的算法, 将行人检测的精度一次又一次地提高, 向着实用的方向不断迈进。本节将对其中最具有代表性的几类算法做一个初步的介绍。

^①张艳敏. 基于机器学习的行人检测[D]. 石家庄: 河北师范大学, 2012.

第一类算法是称为基于整体特征的算法。这种算法把人看成一个不可分的整体,通常用矩形的包围盒来表示行人。对于一张给定的图片,先对其进行特征提取,如小波特征、hog 特征等,再直接用分类器,如支持向量机等,进行分类。这类方法的优点是整个框架比较简单,易于实现。如果提取的特征能有效地描述行人的特点,那么也可以取得很好的效果。但由于将行人作为一个整体来处理,忽略了人体的非刚性,所以这类方法在处理遮挡、不同姿势、视角等方面显得不够灵活。在这类方法中,能否选取有效的特征是成败的关键。由于行人的服饰各异,颜色、灰度和纹理都难以成为稳定的特征,而人体的轮廓恰好受以上种种因素的干扰最小,从而也是最稳定的一种特征。因而在基于整体特征的算法中,通常把研究重心放在如何更有效地提取以及表示人体的轮廓上。

第二类方法称为基于多部位的方法。该类方法以检测部位为基础,在得到了各个部位的检测结果后,再分析各部位的相互约束关系来得到最终的结果。这类方法的优点在于能更好地处理遮挡以及行人姿势的多样性。这类方法的主要问题则在于,如何定义部位以及如何整合来自多个部位检测器的信息。

第三类方法是基于多视角的方法。在人脸检测领域,侧面人脸和正面人脸通常是分开训练,来减小类内变化,简化训练难度。受到这个思路的启发,也有研究者将不同视角的行人分别处理,以便更好地处理不同视角下行人外观不同的问题。这类方法中的主要研究问题则是:如何不依赖于人的标注自动地分视角学习?如何尽量利用不同视角下的相似性来减小计算量?需要说明的是,以上的分类并不相互排斥,例如在基于多部位的方法中,就可以把每一个部位看成一个整体,从而利用第一类的方法来描述各个部位。又比如,第二类方法和第三类方法也可以结合起来,把每个视角下的每个部位分别训练,成为基于多部位多视角的方法。事实上,这三种方法正是为了处理在第一章所列举的行人检测的七大难题而提出的。基于整体特征的方法处理行人的特征表示问题,致力于在不同的衣着、身材、背景、光照的条件下,获得稳定的特征。基于多部位的方法处理行人的姿势、遮挡以及非刚性。基于多视角的问题处理人体的正面及侧面在外观上的差异。因此,我们这样分类便是为了能更清楚地说明行人检测中最最重要的问题,以及针对这些问题的解决方案。

7.4.2 基于整体特征的检测方法

1. 小波特征

在1997年,Oren提出了用小波提取特征,再使用支持向量机来做分类的思想。虽然在现在看来,该算法的效果还远远算不上理想,但是在当时,这一算法不仅取得了领先的效果,更为重要的是:首次将使用机器学习的思想引入了行人检测领域,通过对训练样本的学习自动地建立分类模型,从而摆脱了原先手工建立模型的方法。以此为标志,人体检测正式进入了机器学习的时代。而且这一算法确立了一个基本的框架,理解这一方法将帮助我们更好地理解后来出现的更为复杂的算法。下面我们将对这一方法做一概述。

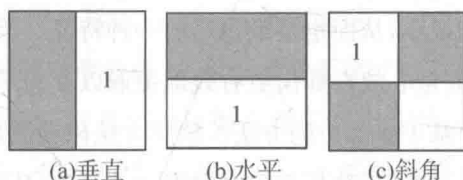


图 7-14 三种类型的小波特征

Oren 首先准备了 64×128 的图片来准备行人样本。然后用水平、竖直、斜角 3 个方向的小波在不同尺度上提取特征(如图 7-14),限于当时的计算机的计算能力比较有限,该作者又从上千个小波系数中选取了 29 个系数作为特征,最后再用支持向量机来训练分类器。此时还需解决的另一大问题是:如何定义负样本? 怎样的样本才是典型的非行人样本? 这里作者采用了 bootstrap 的方法:首先在不包含行人的图片中随机地选取一些样本作为初始的负样本,训练出一个初始的分类器后,再把该分类器分错的负样本加入初始的负样本,重新训练。这样便可以有效地选取有代表性的负样本,提升分类器的效率。最终,Oren 获得了当时领先的效果。在 $1/15\,000$ 的误报率下,获得了 81.6% 的检测率。总的来说,Oren 的方法有两个关键之处:其一,他证明了通过提取一些看似简单的特征,再训练分类器加以分类是完全可行的方案;其二,他证明了使用 bootstrap 可以有效地筛选出最有代表性的负样本,从而大大提高分类器的精度。在 Oren 之后,又有不少科研人员循着他的足迹,提出了更为有效的算法,如 hog 特征和 shapelet 特征。

2. 边缘模板

如果说行人的哪个特征最为直观? 那么显然答案是轮廓。在 1999 年,Gavrila 提出了一个实时的行人检测算法并将之应用到了智能车辆上。这个算法的出发点非常直观:直接使用行人的边缘模板来对整张图片做模板匹配,只

要匹配度大于一个阈值,就认为是行人。边缘模板的例子如图 7-15。



图7-15 边缘模板示意图

对于图像中的一个给定位置,模板与图像的距离可以用 chamfer 距离来计算。

$$D_{\text{chamfer}}(T, D) = \frac{1}{|T|} \sum_{i \in T} d_i(t) \quad (7-1)$$

其中, $|T|$ 表示模板中的特征数量,也就是边缘点的数量。 $d_i(t)$ 表示在图像中与点 t 最近的点的距离。chamfer 距离用 distancetransform 来计算。

接下来的一个问题是:一个模板肯定不足以表达行人各式各样的外观。所以该作者使用了整整 1000 个模板来匹配。如果采取最简单的算法,在图像的每一个位置上都用所有模板去匹配,那么其计算量是非常惊人的,所以 Gavrilu 利用类似 k-means 的算法对模板进行层次聚类,得到一个层次化的树形模板,在每个节点上,仅选择与待测图像最接近的模板进行搜索,这样就大大简化了计算。树形模板的示意图如图 7-16。

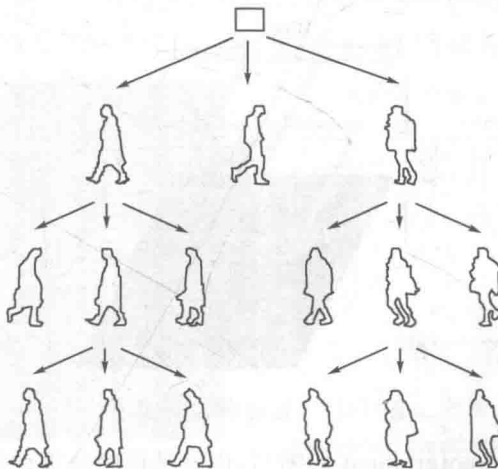


图 7-16 行人的树形模板示意图

除此之外,为了达到实时的处理效果,Gavrila 还提出了从粗到细的搜索方式以及硬件加速。最终该算法在每张图片 2 个左右误报下,检测率为 75%~85%,运行速度为每秒 1~5 帧。总的来说,据我们所知,这是行人检测领域第一个达到实时的基于边缘的学习算法。然而该算法的缺点也很明显:其一是需要大量的人力来标注模板;其二是边缘模板依赖于二值化的边缘提取,在行人的边缘不那么显著的情况下,可能会出现漏检。而且该算法只利用了边缘的强度信息而忽略角度信息;其三是大量的重复计算,即便是使用了树形的层次化模板,但是树的不同层次的边缘模板仍然有很大的相似性,许多计算还是重复的。

3. hog 特征

在 2005 年 Dalal 提出了基于 hog 特征的算法。其算法框架与上文 Oren 的并无本质不同。唯一的差别只有两点:其一是由于计算机性能的提高,Dalal 使用了上千维的特征,而非像 Oren 那样仅仅选出 29 维;其二是 Dalal 使用了表达能力更强的 hog 特征。但令人惊叹的是,这个看似简单的算法竟然取得了一流的效果,在 Mit 行人库上表现近乎完美,以至于 Dalal 不得不自己创建了一个难度更高的 Inria 行人库。目前该行人库已成为了行人检测领域的标准测试数据。在 Inria 行人库上,Dalal 在万分之一的误报率下取得了近 90% 的检测率。即使在今天,该算法依然是行人检测领域最优秀的算法之一。下面我们就来介绍一下“神奇”的 hog 特征。

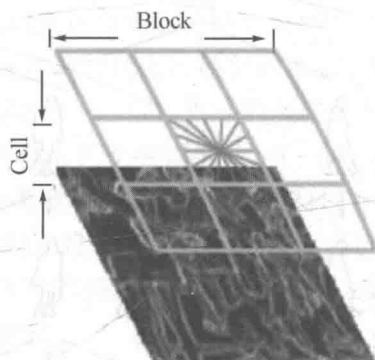


图 7-17 hog 特征示意图

hog 是 histogram of gradient 的缩写,也就是梯度分布直方图。其计算过程如下:首先,将输入图片分成若干块,每块再分成若干格子,对每个格子统计其

中所有像素的梯度值在各个方向上的分布,得到特征向量;其次,把一个块内所有格子的特征向量串联起来便得到了该块的特征向量。图 7-17 给出了 hog 的示意图。

获得 hog 特征后,Dalal 进一步再把所有块的特征向量串联起来作为一个样本的特征向量。由于 hog 非常有效,而且特征的维数也达到了三千维以上,Dalal 的实验表明,即使用线性支持向量机来分类,其效果也仅略逊于非线性的支持向量机。hog 特征的有效性主要来源它能够很好地刻画人体的边缘特征,对光照变化和小量的偏移也并不敏感。事实上 hog 的特征并不是全新的概念。在 hog 出现之前,类似的特征已经有了。Lowe 提出的 sift 特征在计算上和 hog 仅有微小差异,所不同的是,sift 特征仅在一张图片中的兴趣点上提取,其应用范围主要为图像匹配。而 Dalal 的贡献在于将其成功地移植到了物体检测上,不需要依赖于兴趣点检测,并且通过实验证明了基于梯度分布图的特征不仅仅适用于图像匹配,也能很好地适用于物体检测。

4. Edgelet 特征

可以说,边缘是人体最显而易见又最稳定的特征,问题在于人的身材、姿势各有差异,如何才能利用好这些特征呢?与 Gavrilu 为人的整体定义边缘模板不同,BoWu 最先提出了 edgelet,每一个 edgelet 就是一条小边,描述了人体的某个部位的轮廓,然后再用 boosting 算法筛选出最有效的一组 edgelet 来描述人的整体。相较 Gavrilu 的边缘模板而言,该方案不需要人工标注,而且避免了相似模板之间的重复的计算。下面我们就来看一看该方法的主要内容。

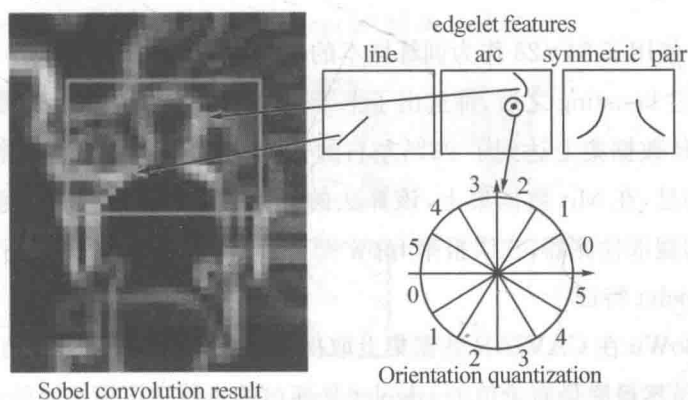


图 7-18 edgelet 示意图

如图 7-18 所示, BoWu 定义了直线型、弧型和对称型三种 edgelet。直观地看, 一个 edgelet 就是一条具有一定形状和位置的“小边”, 由一组边缘点构成。给定一张图片的某个位置, 根据该图片在该位置上是否具有与该 edgelet 形状类似的边缘, 便可以得到一定的响应值。图片中的边缘与 edgelet 越是相似, 那么响应值就越高。edgelet 在一张图片中某个位置 (x, y) 上的响应可用下式计算。

$$S(x, y) = \frac{1}{K} \sum_{i=1}^K I_e(x+u_i, y+v_i) | \langle N_e(x+u_i, y+v_i), n_i \rangle | \quad (7-2)$$

其中, K 是该 edgelet 中点的数量。 (u_i, v_i) 是该点在 edgelet 中的位置。

$I_e(x+u_i, y+v_i)$ 是图片中对应该点的边缘强度, $| \langle N_e(x+u_i, y+v_i), n_i \rangle |$ 表示图片中 $(x+u_i, y+v_i)$ 点的梯度的法矢量和 edgelet 中该点的法矢量的内积。

从 edgelet 的定义中, 我们就可以看出 edgelet 相对于边缘模板的“聪明”之处: 首先, edgelet 不需要对图片二值化, 只要用 sobel 算子计算边缘即可, 这样受光照的影响就比较小; 其次, edgelet 同时考虑了边缘的强度和方向, 更有效地利用了信息。对于背景中那些与 edgelet 在形状上恰好比较相似的边缘, 我们还可以用边缘的方向信息将之排除; 再次, 每个 edgelet 仅负责人体的一小块区域, 而不是像边缘模板那样一口气描述整个人体。这样不但更加灵活, 而且也降低了计算量; 最后, edgelet 是用一定的规则生成, 再用机器学习算法自动地筛选, 不需要像边缘模板那样手动地标注, 大大增加了算法的智能, 减少了人工的干预。

BoWu 使用了 54×28 作为训练样本的大小, 在其中共定义了近 80 多万个 edgelet, 经过 boosting 之后, 筛选出了上千个最有价值的 edgelet。最后该算法在 CAVIAR 数据集上达到了 92% 左右的检测率, 其误报率仅为每帧 0.4 个。值得一提的是, 在 Mit 数据集上, 该算法的效果远远超过了基于小波的方法。例如同样的腿部检测器, 在误报率 $fppw=10^{-5}$ 时, 该方法的检测要高出 15%。

5. Shapelet 特征

虽然 BoWu 在 CAVIAR 数据集上取得了很好的效果, 但是该方法也有其明显不足: 虽然最终最有价值的 edgelet 是通过的机器学习筛选出的, 但是每个单独的 edgelet 是手动设定的, 例如 $1/4$ 圆弧形的 edgelet, 但这一定符合人体

上的曲线吗? 还有一些复杂的曲线本身就是难以用手工设定的 edgelet 来表达的。那么能否用机器学习的方法来自动学习人体的线条呢?

答案当然是能! Sabzmeydani 于 2007 年提出了 shapelet 的算法, 利用机器学习的方法来自动地生成特征, 该方法相较 hog 将误报率进一步降低了整整 10 倍! 以往的算法往往只是利用机器学习来设定分类器的参数或者选择特征, 而在 shapelet 算法中, 特征本身就是通过机器学习得来的, 是机器学习在行人检测中更为彻底与深入的应用。该算法的流程如下。

输入: 训练样本集

对每张训练样本提取 lowlevelfeature;

用 boosting 来生成 shapelet 特征;

对训练好的 shapelet 再使用 boosting, 训练出最终的强分类器。

其中最为关键的流程就是从 lowlevel feature 到 shapelet 这一步。该作者定义的 lowlevelfeature 为 0 度、45 度、90 度和 135 度四个方向上的边缘(图 7-19)。对一个小窗口内所有 lowlevelfeature 用 boosting 进行筛选之后便可以得到一个 shapelet 特征。



图 7-19 Shapelet 的底层特征

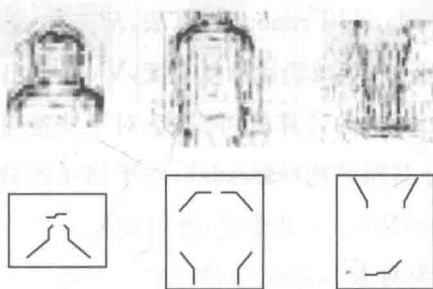


图 7-20 shapelet 和 edgelet 的分别

在头肩, 躯干和腿部选取的特征

在图 7-20 中,我们比较了 edgelet 和 shapelet 在头肩、躯干、腿部三个部位中训练出的特征。从图中我们不难看出,比起 edgelet, shapelet 更能“贴合”人体曲线,更好地描述人体特征。在非常困难复杂的 Inria 数据集上,该算法达到了 90% 的检测率,相应的误报率每十万个窗口一个误报,如果每张图片上扫描 4000 个窗口,那么相当于每张图片 0.04 个误报,超越了以往所有的算法。

7.4.3 基于 boostedcascade 的物体检测

在上一节,我们介绍了行人检测领域的几大研究方向。但事实上,以上的方法往往面临着相同的难题:例如在 edgelet 和 shapelet 算法中,潜在的特征成千上万,我们应该如何选择相应的特征呢?此外,从实用角度上看,检测的速度必须越快越好。一张图片中的大部分区域都不包含行人,如何利用这一点来加速检测呢?本节要介绍的正是对这两个问题的一个经典解决方案,我们称之为 boostedcascade。该方法最初由 Viola 提出,将当时的人脸检测速度提高 10 倍以上,成为之后人脸检测领域的一大经典算法框架,围绕该算法框架展开了大量的研究。在行人检测领域,该算法框架也同样十分适用。上一节介绍的各种算法中,有近一半的算法采用或借鉴了该框架。而在下一节中我们将讨论对该算法做出的改进。因此,在这一节我们有必要先详细地讨论 boostedcascade 算法框架。

在物体检测领域有着几大难题:其一是如何选取特征来表示物体;其二是如何尽量减轻特征计算的复杂度;其三,对于图片中大部分不含物体的区域,如何能快速排除? Viola 提出的 boostedcascade 算法恰恰很好地解决了这三大难题。对于特征选取,Viola 采用 adaboost 算法,从一个过完备的特征集合中自动地选择最有效的特征;对于减轻计算复杂度,Viola 提出了矩形特征与积分图片,每个特征只需几次简单的计算就可完成;对于快速排除不含物体的区域,Viola 提出了级联的分类器结构(cascade),对于简单的背景区域,只需少许计算即可排除。

该算法主要可分为两步:

第一步是利用 boosting 来选择一部分矩形特征构成强分类器;

第二步则是利用 bootstrap 不断选择出最有代表性的负样本,来训练出一

个级联的分类器。

该算法框架的结构可用图 7-21 表示。

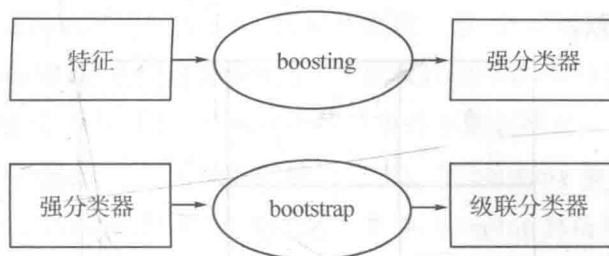


图 7-21 boostedcascade 算法的流程

本节将依次对积分图片和矩形特征、boosting 算法以及级联分类器进行说明，并介绍这个算法在行人检测中的常见应用。

1. 积分图片与矩形特征

由于人脸具有很强的区域特性，例如嘴部，眼睛等部位的颜色总是比周围更深，针对人脸的这个特性，Viola 提出了类似 haar 小波的矩形特征。

图 7-22 中是 Viola 等提出的几种矩形特征。这些特征的计算方法为黑色区域内的像素平均值减去白色区域内的像素平均值。

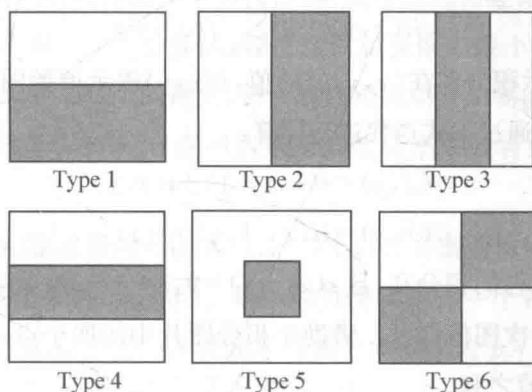


图 7-22 haar 特征例子

由于事先我们并不知道哪些位置、类型和大小的矩形特征会最有用，所以我们不得不穷举所有矩形特征。这个数量是非常惊人的，以 24×24 大小的检测框为例，其中包含的矩形特征共有 180 000 多个！即使通过 boosting 算法选出最有代表性的上千个特征，其数量依然十分庞大。针对这个问题，Viola 第一

次引入了“积分图”(Integral Image)的概念。这使得矩形特征的计算变得非常容易,不再需要访问矩形特征内的所有像素,无论矩形特征的面积大小是多少,计算量都是常数。

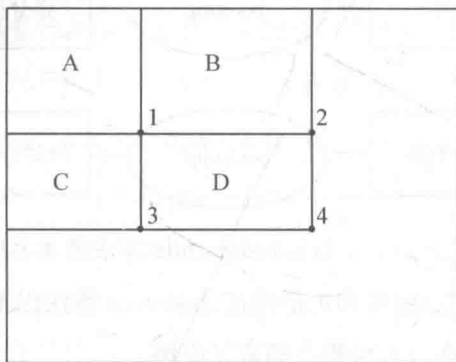


图 7-23 积分图

图 7-23 积分图元素值计算, D 中所有的像素值之和可以用 $ii(4) + ii(1) - ii(2) - ii(3)$ 计算。

积分图片的示意图如图 7-23 所示。积分图中坐标点 (x, y) 处的值定义为原图中左上角的像素值之和:

$$ii(x, y) = \sum_{x' \leq x, y' \leq y} i(x', y') \quad (7-3)$$

其中, $ii(x, y)$ 表示积分图在 (x, y) 处的值, $i(x, y)$ 表示原始图像中对应位置的像素值。 $ii(x, y)$ 通过下式迭代进行计算:

$$s(x, y) = s(x, y-1) + i(x, y)$$

$$ii(x, y) = ii(x-1, y) + s(x, y) \quad (7-4)$$

其中, $s(x, y)$ 表示行的积分和, 且 $s(x, -1) = 0$, $ii(-, y) = 0$ 。求一幅图像的积分和, 只需遍历一次图像即可。借助于积分图片中的四个点, 便可以计算任意矩形中所有像素值之和。

举例来说, 图 7-22 中由两个矩形构成的特征, 其像素和之差可通过访问积分图片中的六个点完成。由三个矩形构成的特征可以通过访问八个点求得; 由四个矩形构成的特征可以通过访问九个点求得。

2. adaboost 算法

很多时候我们知道某些特征会有一定价值, 但是每一个特征单独地使用都

不会产生很好的效果。例如,两只足球队打比赛,球队的实力、近期的表现、主客场等因素都会影响比赛结果,那么如何才能综合这些因素来预测比赛呢?在物体检测中这样的问题也很常见,例如从直观上想,haar 特征肯定是有用的,例如眉毛比周围皮肤的颜色要深就是一个比较好的特征,但这个特征并非在所有人脸上都出现,反过来说,出现这个特征也并不能说明就是人脸。boosting 算法为这类问题提供了一个良好的解决方案。在 boosting 算法中,那些有一定价值却不必然成立的规则都被称为弱分类器,boosting 算法可以将这些弱分类器组合起来,构成一个强分类器。事实上,只要这些弱分类器比随机猜测好上一些,那么最终的强分类器就可以达到任意精度。boosting 的另一优点就是它在很多应用中都显示出了良好的泛化能力。

boosting 算法中最常见的是 adaboost 算法,由 Freund 和 Schapire 在 1995 年提出,并由 Viola 初次引入人脸检测。Viola 的算法是在离散 adaboost(discrete adaboost)基础上发展而来,它能从一个庞大的矩形特征集中选择很小的一部分关键的特征(几百至几千),从而产生一个极其有效的分类器。

表 7-6 是 Viola 所采用的 adaboost 的流程。算法关键部分在于权值更新,更新权值会增大当前被分错样本的权值,从而将注意力逐渐集中在难样本上。本质上,adaboost 是一个贪心算法,每次选取加权误差最小的弱分类器。在训练的过程中,随着训练难度的增大,每一轮选出的弱分类器的误差将逐渐变大,而弱分类器的权值相应变小,直到达到期望的误报率或者最小加权误差达到 0.5。

Viola 采用的是离散的 adaboost,即每个弱分类器的输出是离散值,采用的形式是单阈值的分类器(stump)。每个弱分类器针对一个特征 $f_j(x)$,为该特征确定一个门限值 θ_j ,以及一个指示不等式方向的参数 p_j :

$$h_j(x) = \begin{cases} 1, & \text{if } p_j f_j(x) < p_j \theta_j \\ -1 & \text{otherwise} \end{cases} \quad (7-5)$$

强分类器由一组弱分类器的线性组合而成。

表 7-6 Adaboost 算法流程

给定样本图片 $(x_1, y_1), \dots, (x_n, y_n)$ 其中 $y_i = \{-1, 1\}$ 分别对应负样本和正样本

初始化样本权值 $w_{1,i} = \frac{1}{2m}, \frac{1}{2n}$, m 是正样本数量, n 是负样本的数量

$For t=1, \dots, T$ (弱分类器数量):

归一化权值, $w_{t,i} = \frac{w_{t,i}}{\sum_{j=1}^n w_{t,j}}$, 使得权值 w_t 可以表示概率分布

对于每个特征 j , 训练一个弱分类器 h_j , h_j 被限制只使用一个特征。误差估计是根据权值 w_t 计算: $\epsilon_t = \sum_i w_t (h_j(x_i) \neq y_i)$

根据具有最小加权误差 ϵ_t 的弱分类器 h_t .

更新样本权值: $w_{t+1,i} = w_{t,i} (\beta^{e_{t,i}})$,

其中, $e_{t,i} = \begin{cases} 1, & h_t(x_i) \neq y_i \\ 0, & h_t(x_i) = y_i \end{cases}$

$h_t(x_i) \neq y_i, \beta = \frac{\epsilon_t}{1 - \epsilon_t}$

最终强分类器输出为:

$H(x) = \begin{cases} 1, & \sum_{t=1}^T \alpha_t h_t(x) \geq \sum_{t=1}^T \alpha_t \\ 0, & \text{otherwise} \end{cases}$

otherwise, 其中, $\alpha_t = \frac{1}{2} \log \frac{1}{\beta_t}$

3. Cascade 级联分类器

级联分类器(cascade)是一种链式结构的分类器,由分类能力逐渐加强的一组分类器级联而成,可用来大大加快检测的效率。级联分类器的特点在于,一旦某一级分类器排除了某个样本,那么该样本就被马上判为负,不再进一步地处理。一般来说,最初的几级分类器只负责排除简单的负样本,往往只需少量的特征参与计算。越是靠后的分类器,则分类能力越强,负责排除与人脸非常近似的负样本。级联分类器的每一级都使用 adaboost 算法训练,保证能让大部分的人脸样本通过(如 99.9%),而每级都排除一部分非人脸区域(如

50%)。这个结构可以尽可能快地排除背景区域,从而节约出时间用于对那些,更像目标的区域进行计算。该级联的检测器的结构如图 7-24 所示。

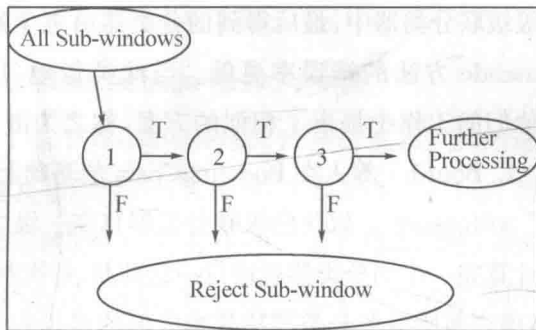


图 7-24 cascade 的结构

训练级联分类器的负样本用 bootstrap 获得。可让前一级的分类器在从不含人脸的背景图片中进行扫描,选取前一级无法正确分类的负样本将作为下一级的训练样本。

4. boostedcascade 的改进

boostedcascade 算法一经提出立刻获得了高度的关注,大量的论文对其进行研究 with 改进。针对 haar 特征,Lienhart 提出了斜角方向的 haar 特征,以更好地处理侧脸;Levi 则提出了 eoh 特征,大大提高了系统的泛化能力,可在小训练样本下得到更好的结果。针对弱分类器,BoWu 提出了查找表来替代 stump,更好地利用了特征,从而不仅提高了检测率,还大大降低了特征的数量。

针对 boosting 算法,BoWu 还提出了与查找表配合使用的实值 boosting,增强了弱分类器的表达能力;XinwenHou 则提出非对称损失的 boosting。在传统的 adaboost 中,正负样本的重要性是相同的,但在检测问题中,正样本的重要性往往高过负样本,宁可出现误报也尽量不能出现漏检,非对称损失的 boosting 可进一步提高检测精度。

针对 cascade 的学习算法,StanZ. Li 提出了一种基于 FloatBoost 的多视角的人脸检测算法。该算法充分考虑了选出的弱分类器可能存在的冗余性,利用 FloatBoost 排除不必要的弱分类器。同基于 adaboost 的算法相比,它能在提高人脸检测速度的同时提高检测的精度。值得一提的是,它还是第一个实时的

多角度人脸检测系统。R. Xiao 等人在 cascade 学习结构上提出了改进,提出 BoostingChain。在这个算法中,作者引入一个链式结构把级联分类器的历史信息传播到下一级级联分类器中,最后得到的分类器由更少的弱分类器组成,并且比原先的 cascade 方法的错误率更低。与此类似地,B. Wu, C. Huang (2004)等人也在他们的工作中提出了相似的方案,称之为嵌套的级联分类器(nestedcascade)。L. Bourdev 等人在 BoostingChain 的基础上进一步改进 cascade 构造,提出了 softcascade。以往的级联分类器算完一级才做一次判断,而 softcascade 则每计算完一个弱分类器就进行一次判断,每次判断不光考虑当前的特征响应,还考虑之前所有弱分类器特征响应值的累积和。在这种结构下,分类器的历史信息得以保存,目标物体也不会因为一个分类器上的响应值低而被排除。作者指出该架构适用于物体类内变化比较大的情况,例如将正面与侧面的人脸一起训练。

5. boostedcascade 在行人检测中的应用

boostedcascade 算法不仅在人脸检测领域大行其道,在行人检测领域也非常流行。举例来说,Viola 在 2003 年提出同时在灰度图和帧差图中来提取 haar 特征,既能描述人体的外观又可描述人体的运动。QiangZhu 则用 hog 特征代替 haar 特征,用线性支持向量机代替 stump 作为弱分类器,并且利用积分图片来加速 hog 特征的计算,从而将 Dalal 的 hog 算法的运行速度提高了几十倍,并且在分类精度上基本没有损失。Laptev 做了与 QiangZhu 相似的改进,区别之处在于 Laptev 利用投影来对 hog 特征降维,提高了训练速度。OncelTuzel 则提出了将斜方差矩阵用于行人检测,并利用流形和 logitboost 来进行学习,取得了比 hog 更好的效果,在万分之一的误报率下($fppw=10^{-4}$),可达到 93% 的检测率。遗憾的是,尽管使用了级联分类器,由于斜方差矩阵的计算过于耗时,该算法每秒钟尽可检测 3000 个图片窗口,与 Dalal 的 hog 算法没有明显区别。

此外,在第二节中我们介绍的 edgelet 特征也使用了该框架。例如从 80 多万个潜在的 edgelet 中选取一小部分便是利用了 adaboost,最终的分类器也是级联结构的分类器。值得一提的是,通过 CBT,BoWu 更是利用 edgelet 特征在 Inria 测试集上超越了 hog 算法的精度。

Sabzmeydani 提出的 shapelet 方法虽然没有完全采用该框架,但是也采用了 boosting 来做特征选取,也采用了积分图片来加速运算。除了没有使用级联分类器之外,其他方面并无本质不同。

7.4.4 基于 boostedcascade 的行人检测

上一节介绍了基于 boostedcascade 的算法框架。虽然这一框架具有种种优点,但却并不能直接应用于行人检测,主要原因在于 haar 特征适合于描述显著的区域特征,如眼睛、嘴部、眉毛,但并不适合描述边缘,因此也就并不完全适用于行人检测。在本节中,我们将从特征的选取、弱分类器的设计、boosting 算法三个方面对其加以改进。在特征选取上,我们提出了 hog 特征的简化版本 shog(simplehog),并且将 shog 特征与 haar 特征结合起来使用,这样既能利用 hog 特征获得良好的精度,又能利用 haar 特征提高检测速度。在 boosting 算法上,我们采用了实值 adaboost 来取代离散的 adaboost,从而更有效率地利用了特征。

在弱分类器上,我们提出了先用加权的 fisher 判别寻找最佳的投影方向,把多维的 shog 特征将为一维,再用查找表(lookup table)进行分类,大大缩短了训练时间,提高了分类器的精度。

本节将对这三个方面的改进逐一说明并给出在 Inria 数据集上的实验结果。

1. 特征的改进

通过前一节的介绍,我们已经知道 Dalal 提出的 hog 特征在行人检测上的优异表现。但是 Dalal 提出的算法也存在速度过慢的问题,即使用线性的支持向量机,面对每个图片窗口中三千多维的特征向量也显得力不从心。根据 Dalal 的实验,如果每张图片检测 4000 个窗口,那么还是需要 1 秒的运行时间。显然这是无法满足实时的要求的。之后 QiangZhu 将 hog 取代了 haar 特征,利用 boostedcascade 的算法框架将检测速度大大提升。但是这样的速度还是无法达到实时的要求。受到 BoWu 将 haar 特征与 edgelet 特征结合起来的启发,我们提出将 haar 特征与 hog 特征结合起来,同时我们提出将 hog 特征简化为 shog 特征,将其计算量降低了 75%。

hog 特征最初来源于 sift 特征, Dalal 将之引入行人检测。但问题在于, 为什么当初 sift 要将每一块分为多个 2×2 个单元呢? 这是因为 Lowe 发现, 划分得细致可以使特征向量更具有独特性, 在图像匹配的时候, 特征越独特就越不容易发生误匹配。但过度的划分也有坏处, 一方面是增加了计算量, 另一方面也使得特征过于独特, 以至于对于定位误差、光照变化、物体的形状变化等过于敏感。而 2×2 的划分恰恰在两个因素之间做到了最好的平衡。我们的问题在于: 2×2 的划分一定适合行人检测吗? 既然 boosting 算法可以将多个特征的信息加以融合, 那么每一个单独的 hog 特征是否有必要具有很强的独特性? 为此, 我们提出了简化版本的 hog 特征, 也就是不再对每一个块进行细分, 直接在一整块中计算梯度分布图。

下面我们将给出我们的实验结果。我们采用的数据集是 Inria 数据集, 训练正样本为 2416 张 32×64 的行人图片, 负样本也是 2416 张, 从 1218 张不包含行人的背景图片用 bootstrap 的方式获得。如无特别的说明, 下面的实验都是在该数据集上进行。在算法方面, 我们采用了 realadaboost 以及基于 fisher 判别投影的查找表作为弱分类器。

实验一: 在此实验中我们说明结合 haar 和 shog 的优势, 如图 7-25。

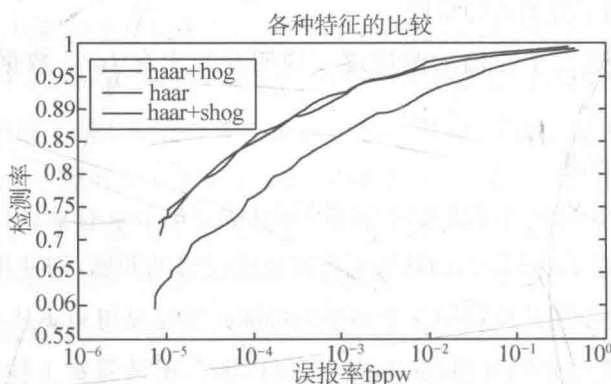


图 7-25 各种特征的比较

从图 7-25 中我们可以看出, 在万分之一的误报率下 ($fppw=10^{-4}$), 单独使用 haar 特征的检测率为 77%, 在载入了 hog 特征后, 检测率提高到了 86%; 如果我们用 shog 替代 hog 特征, 我们的检测率基本没有任何变化, 但由于 shog 特征的计算复杂度仅为 hog 的 $1/4$, 我们的训练速度可提高 2 倍, 检测速度提

高 30%。最终,在 1.8 GHz CPU 的计算机上,我们的基于 haar 和 shog 的算法每秒可以检测 6 万个图片窗口,对于 640×480 的图片上可达到每秒 2 帧,比 Dalal 的每秒检测 4000 个图片窗口快了近 15 倍。

2. boosting 算法的改进

在提出了 discreteadaboost 之后, Schapire 等又提出了实值 adaboost (realadaboost)。实值 adaboost 和离散 adaboost 相比,最大的区别在于弱分类器的输出不再是离散的 $\{-1, 1\}$, 而是一个连续的数域, 越大则说明该弱分类器对该样本为正的置信度越高。这个方法也被称为置信度相关的预测。式 7-6 表示的 stump 类型的分类器只能输出两个值, 所能提供的信息非常有限。根据 Schapir 的理论, 在实值 realadaboost 中理论上最优的分类器为:

$$h(x) = \log \left(\frac{\hat{p}(y=1|x)}{\hat{p}(y=-1|x)} \right) \quad (7-6)$$

为了对上式中的后验概率密度进行估计, 在实际操作中, 我们采取了 BoWu 的查找表 (look-up table)。形式如下:

$$\forall x \in \text{Bin}, h(x) = \frac{1}{2} \ln \left(\frac{w_j^{+1} + \epsilon}{w_j^{-1} + \epsilon} \right) \quad (7-7)$$

其中, w_j^{+1} 和 w_j^{-1} 分别是在特征响应 X 处相应的正、负样本分布的值。从图中易见, 这是一个分段函数, 和 stump 相比 (图 7-26), 查找表提供了更强的分类能力。由于最大程度上利用了特征的信息, 实值 Boosting 可以使用更少的特征达到指定的检测率。

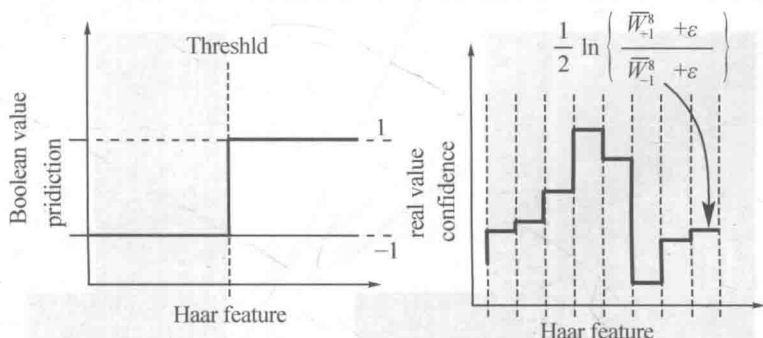


图 7-26 Stump 与 BoWu 提出的查找表的区别

在本实验中,我们比较了 discreteadaboost 与 realadaboost 在使用特征的数量上以及检测精度上的差别。我们使用的特征都是 hog 特征。对于 dis-

creteadaboost, 我们使用 stimp 作为弱分类器; 对于 realadaboost, 我们则采用了查找表作为弱分类器。

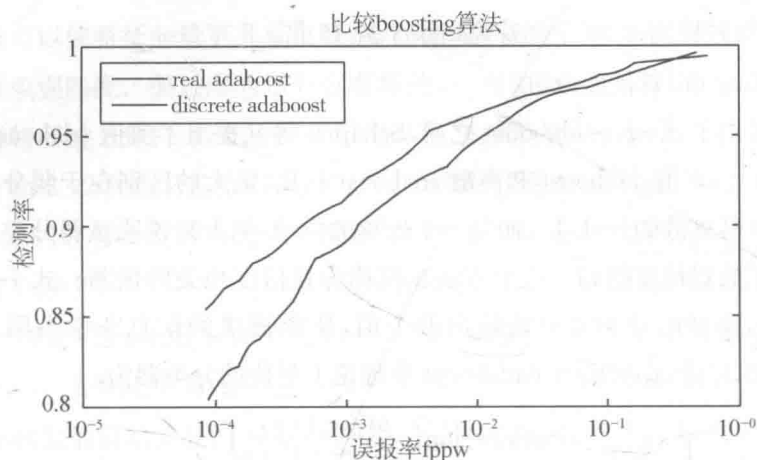


图 7-27 boosting 算法的比较

在图 7-27 中, 我们可以看到在使用了实值 adaboost 之后, 检测精度大大提高。在误报率为万分之一时, realadaboost 的检测率比 discreteadaboost 高出了 5%。与此同时, 使用的特征也大大减少, 在图 7-28 中, 我们分别画出了用两种 boosting 算法训练出来的级联分类器的前三级所使用的特征。其中, 用实值 boosting 训练出来的分类器仅使用了 7 个 hog 特征, 而离散的 boosting 训练出的分类器则使用了 14 个特征, 比前者多了整整一倍。由于实值 boosting 使用了更少的特征, 训练速度和训练速度也都能大为提高。

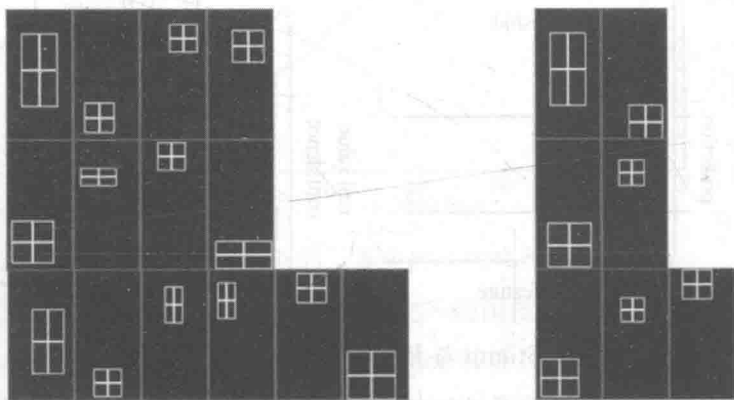


图 7-28 级联分类器的前三级

左图中的是离散 boosting 训练出的分类器。右图中是实值。

3. boosting 训练出的分类器

(1) 弱分类器的设计

在上面两节中,我们已证明了 realadaboost 以及 shog 特征的优势。但是这其中还有一个关键性的问题:realadaboost 需要对后验概率做出估计,查找表虽然可以完成这一任务,但查找表只能处理一维的数据,而我们的 shog 特征则是多维的。那么怎样才能把 shog 特征作为一维呢?

QiangZhu 利用线性支持向量机做弱分类器,但由于支持向量机的训练时间过长,QiangZhu 不得不随机选取 200 多个 hog 特征进行训练;Laptev 提出利用加权的 fisher 判断寻找最佳的投影方向,将 hog 特征简化为一维,再用 stump 分类。投影方向的选取可用下式表达:

$$\begin{aligned}\mu &= \frac{1}{n \sum_{i=1}^n w_i} \sum_{i=1}^n w_i f_i \\ S &= \frac{1}{(n-1) \sum_{i=1}^n w_i^2} \sum_{i=1}^n w_i^2 (f_i - \mu)(f_i - \mu)^T \\ w &= (S^+ + S^-)^{-1} (\mu^+ - \mu^-)\end{aligned}\quad (7-8)$$

其中, w_i 是第 i 个样本的权重, f_i 是样本 i 的特征向量。 μ 是加权重心, S 是加权协方差矩阵, S^+ 和 S^- 分别表示正负样本的加权协方差矩阵, w 是投影方向。

我们进一步改进了 Laptev 的方法,在投影之后,利用查找表作为分类器,更充分地使用了 hog 特征的信息。

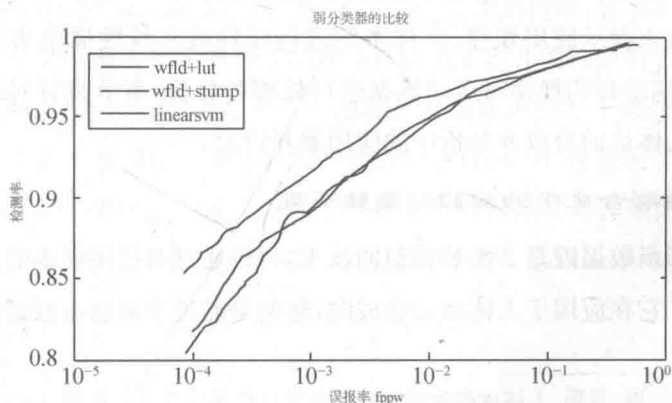


图 7-29 比较各种弱分类器

在图 7-29 中, wfld+lut 代表先用加权 fisher 投影再用查找表做弱分类器, wfld+stump 表示投影后用 stump 做弱分类器, linearsvm 则表示用线性 svm 做弱分类器。

可以看到我们的提出的方法精度最高。在误报率为万分之一时, 我们的检测率为 86%, 而 wfld+stump 的方法为 81%, linearsvm 的方法为 82% (这是在我们的实现中该算法的精度, 由于实现细节的不同, 在 QiangZhu 的论文中, 他们的检测率稍高)。此外, 我们的方法在训练速度上也更快, 只需要 8 小时, 而如果用线性支持向量机, 那么训练的时间需要 3~4 天。这要归功于 fisher 投影的速度要远远快于线性支持向量机。

(2) 小结

我们将 boostedcascade 应用到了行人检测中, 评估了各种特征的特性, 提出了 hog 的简化版本 shog, 并将 hog 特征的描述能力与 haar 特征简单快速的优点结合起来, 既能避免原始的 hog 算法计算复杂度高, 不适合实时检测的缺点, 又能弥补 haar 特征过于简单, 精度不大的缺陷。除此之外, 我们提出利用加权的 fisher 判别来做投影, 将 hog 特征简化为一维, 利用查找表做弱分类器; 并使用 realadaboost 取代原先的 discreteadaboost, 既简化了计算, 又提高了性能, 达到了更高的检测精度。

7.5 机器学习在运动合成与分析中的应用^①

从本质上讲, 人体运动合成与分析中使用的机器学习技术都是通过对已有样本的分析来揭示底层规律、在样本间进行线性或非线性插值来合成新的运动, 或者利用得到的规律对运动数据进行处理和分析, 本节将针对各类机器学习技术在人体运动合成和分析中的应用展开讨论。

7.5.1 运动合成中的回归与函数逼近

回归与函数逼近是 2 类较相似的技术, 可以通过对已用样本的分析来合成新的样本。它在应用于人体运动合成时, 通常是通过学习已有数据得到参数化

^①刘更代, 潘志庚, 程熙. 人体运动合成中的机器学习技术综述[J]. 计算机辅助设计与图形学学报, 2010, 22(9): 1619-1627.

模型,以此来对运动进行控制。Rose 等提出一种多维运动插值的方法,将运动的内容类比为动词,将风格类比为副词,即一个二维参数空间。他们采用径向基函数(Radial Basis Functions, RBF)对若干具有相同内容和不同风格的运动进行插值,生成多维“副词”空间。该方法简单易用,只需输入几段相似运动就可以得到令人满意的效果。2001 年 Rose 等又使用 RBF 插值来求解逆运动学问题,通过在样本间插值得到满足约束的虚拟人位姿。但是该方法对输入样本依赖性强,且只适用于小范围运动。这类方法虽然实时性好,但是适用性较差。通过在样本间插值还可以实现对运动风格的变换。Torresani 等实现了一个基于运动混合的风格化运动控制器,通过学习运动混合的权值来合成新的风格化运动,但该方法的风格参数必须人为标注。Wang 等用高斯混合模型对拳击数据进行训练,得到了一个从高维数据到风格参数(包括出拳距离和蹲姿的高度)的映射,可用于合成不同风格的出拳动作。Mukai 等则将运动插值看作是对任意参数空间中遗失数据的预测问题,并采用地统计学(Geostatistic)插值方法来估计运动姿态非相似度与其在对应参数空间中数据点距离之间的映射。Liu 等将机器学习和逆动力学结合起来,首先针对一段运动使用逆动力学和能量函数构建约束方程,然后使用非线性反向优化学习出方程中表征风格的物理参数,最后通过修改这些参数来合成各种风格的运动。

鉴于回归分析具有生成新数据的能力,有人用它来对数据捕获和运动编辑过程进行简化和优化。Cooper 等使用主动学习技术构造了一个可以在线创建运动控制器的方法。主动学习是一种用于实验设计的技术,它并不从数据中挖掘模型,而是动态地告知用户为了构建模型应该采用什么样的数据。Cooper 的系统能够根据任务告诉使用者下一步捕捉什么动作可以最大限度地提高控制器的性能,本质上这也是一种数据回归的方法。Ikemoto 等提出了一个有趣的方法,其用于将小段的运动编辑操作推广到更长的运动或者其他的角色的运动之上,如图 7-30 所示。该方法使用了高斯过程(Gaussian Processes, GP),它与 RBF 一样都是一种基于核的回归方法。



图 7-30 使用 GP 回归实现运动编辑的推广

人体运动数据除了在空间上具有相关性之外,作为一种时序信号,它在时间上也有一定的相关性。有一些方法采用动态模型对时间相关性进行建模,而这类方法可看作是对函数逼近方法的应用。Brand 等使用隐马尔可夫模型 (Hidden Markov Model, HMM) 对舞蹈运动进行建模,并使用隐变量来表示风格参数,学到的风格可用于生成其他运动。Li 等提出了一个两层统计模型,其底层是用线性动态系统 (Linear Dynamic System, LDS) 表达的基本单元,称为 Motion Texton, 高层是这些单元之间的转换关系,用 HMM 表示,训练得到的模型可以用来合成新的运动。LDS 表达了运动在时域上的局部动态特性,而 HMM 则在全局上描述了运动的变化。HSU 等提出了风格转换的概念,通过学习两段运动之间的差别来建立一个线性时不变 (Linear Time Invariant, LTI,) 模型,并以该模型驱动运动风格之间的转换和过渡。Chai 等将基于约束的运动合成看作是求解最大后验问题。从运动捕获数据中学习出一个 LTI 模型作为运动先验,该运动先验和用户指定的约束共同构成一个轨迹优化问题,求解该优化问题即可得到满足用户需求的动画,如图 7-31 所示。除了 LDS 和 LTI 这些线性系统之外,也有人使用非线性系统来对运动数据建模。Wang 等使用高斯过程动态模型 (Gauss Process Dyna Micmodel, GPDM) 对人体运动建模,并合成新的连续运动。GPDM 是一种适用于时序数据的非线性隐变量模型,与一般的隐变量模型不同,GPDM 考虑了输入数据的时间结构。Lau 等叫使用动态贝叶斯网络 (Dynamic Bayesian Networks, DBN) 来对输入的运动样本进行扩展,从少量的样本衍生出大量在空间和时间上都有变化的具有新风格的样本。该技术不仅仅能运用在运动数据上,对图像数据也能胜任。Silva 等

提出了一个风格化运动控制器,该控制器将平衡保持和风格处理结合在一起,采用线性时变(Linear Time Variant, LTV)系统来处理平衡问题,采用捕获运动来提供运动风格,然后将这两部分归纳为一个优化方程,最后通过求解二次规划方程来合成适应新环境的风格化运动。使用动态模型的优势在于考虑了运动在时序上的动态特性,因此合成出的运动连贯、自然,符合视觉经验。

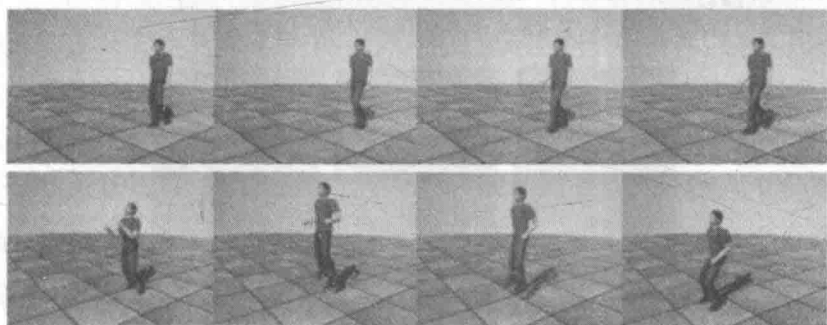


图 7-31 用统计动态模型的运动合成方法

采用回归和函数逼近方法通常需要输入一定量的采样数据,并且训练过程需要较长时间,并不适用于在线游戏和交互式编辑,主要应用于动画的制作和仿真。在运动数据分析方面,Ren 等采用数据驱动的方法对运动数据的真实感进行评价。该方法首先使用统计模型(包括混合高斯模型(Gaussian Mixture Model, GMM), HMM、转换式线性动态模型及朴素贝叶斯模型等)来学习运动数据的内在规律,然后用建立好的统计模型对待评估的数据进行测试,以验证数据的真实程度。

7.5.2 降维在运动合成中的应用

降维通常可以得到数据的内在结构,并且能起到压缩化数据的作用,适用于人体运动这样的高维数据。很多非监督的机器学习技术具有降维功能,常用的有主成分分析(Principal Components Analysis, PCA)和独立成分分析(Independent Components Analysis, ICA)。Glarion 等用 PCA 创建了一个数据驱动的适用于不同步速和步幅的行走引擎。Urtasun 等使用 PCA 对不同速度和风格的行走运动数据进行训练,利用 PCA 系数拟合出的函数合成多种风格的行走运动。李淳芃等将运动数据看作连续函数,首先对数据进行拟合,然后

使用函数 PCA 算法学习走路、上楼梯等运动数据的不同风格,并将其用于合成。Mori 等使用 ICA 来寻找运动数据中的独立特征,然后在 ICA 子空间对运动进行插值,合成新的运动。Shapiro 等则采用 ICA 对运动数据对进行分离,提取运动风格,然后将其传给其他运动。Liu 等的方法与 Shapiro 的类似,使用独立特征子空间分析将运动数据分解为若干子空间,并用其中之一来表征运动风格,实现对风格的调节、传递和混合,如图 7-32 所示。

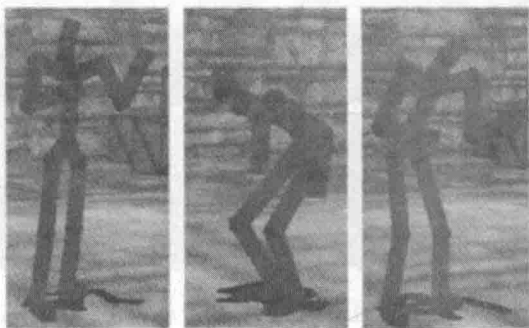


图 7-32 体运动的风格子空间

由于独立特征子空间分析能够表达某些高阶不变特性,对于大多数循环运动它都可以有效地找到风格子空间并对其进行编辑。Shin 等们使用多维尺度变换(Multi Dimensional Scaling, MDS)将运动帧降维,然后在低维空间通过草图勾画的方式直接对运动进行插值、连接等编辑工作,该方法操作简便但不直观。

降维也能够用于辅助求解运动学(Inverse Kinematics, IK)问题,关键在于如何建立有效的参数化模型,使得到的数据能够满足约束。Grochow 等使用一种非线性 PCA 技术—变尺度高斯过程潜在变量模型(Scaled Gaussian Process Latent Variable Model, SGPLVM)来降低数据维数,得到潜在变量空间,然后通过非线性插值来合成新的姿态。SGPLVM 具有极强的泛化能力,只需要少量样本就可以插值出多样的新数据。李淳芃等的方法与 Grochow 的类似,使用自组织映射(Self Organized Map, SOM)对运动数据降维。SOM 可以将相似的高维数据降为低维空间中的相邻数据,得到具有拓扑意义的低维图谱。该方法通过在重新组织后的数据间进行线性插值实现 IK 求解。Tournier

等提出了一个基于主测地线分析(Principal Geodesic Analysis, PGA)的逆运动学求解方法,它是 PCA 在黎曼流形上的扩展,属于流形学习。该方法试图挖掘人体运动数据内的低维流形嵌入,并使用李代数来表示人体关节的旋转角,通过对降维后的数据进行非线性插值求解 IK 问题。由于 PGA 方法具有解析表达形式,因此其计算简便、快速。以上 3 种方法都将 IK 问题转化为低维空间的约束优化问题,这样做的好处是能够显著降低雅可比矩阵的尺寸,同时缩小数值优化的搜索空间,实时求解 IK 问题,满足交互式编辑的需要。

降维技术还可以和传统的 IK 求解方法,如雅可比伪逆结合起来使用。Carvalho 等给出了一种结合低维运动模型和传统 IK 的交互式运动合成方法,该方法使用 PCA 将运动数据降到低维,然后在低维空间求解 IK 优化问题。他们通过简单的链式法则把高维的全身雅可比矩阵分解为降维矩阵和低维雅可比的乘积,从而建立了低维参数空间和高维可视空间的联系。Raunhardt 等们提出了一种类似的框架,不同的是 Carvalho 的运动模型是建立在运动空间的,而 Raunhardt 则是建立在帧空间的。Liu 等对 Carvalho 的方法做了扩展,并提出了一个可以编辑运动风格的逆运动学求解器。首先,使用独立特征子空间分析对运动数据降维;其次,在低维风格空间求解基于雅可比伪逆的 IK,并强制约束低维风格子空间的范数,在合成满足几何约束运动的同时还可以指定风格参数。该方法以拳击运动为例,可以合成击打指定位置的各种勾拳或摆拳风格。由于以上几种方法只需要较少的运动数据作为辅助,因此能够保证运动合成的实时性,可以用于交互式编辑和计算机游戏,但是线性降维方法和稀疏的样本对于那些需要满足多个约束的问题常常效果不佳。

还有一些方法利用降维技术来辅助动力学仿真。Safonova 等使用运动捕获数据训练出一个 PCA 空间,然后在该低维空间运用空间约束优化合成满足约束和物理规律的跳跃动画,该方法可以有效减少参数数量并缩小优化计算时的搜索空间。Ye 等提出了一种基于物理的平衡保持算法,其根据生物力学知识,首先通过逆动力学计算一段捕获数据的内部扭矩,然后将作用于各个关节的扭矩进行 PCA 分解,以在不活跃的坐标窄间(即小特征值对应的特征向量构成的子空间)进行平衡恢复的计算,如图 7-33 所示。该算法是建立在生物力学知识上的,即人体保持平衡时总是使用尽量少的力和能量。

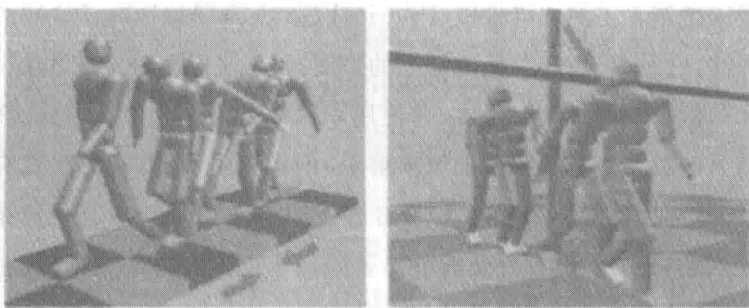


图 7-33 用 PCA 的反应式动画合成

降维也常应用于运动数据处理,因为低维空间通常能够抽象并简化问题。朱登明等使用了类似 Hsu 等的方式建立人体运动的系统,将高维数据映射到低维状态空间,然后在该低维空间中定义了虚拟人位姿的相似性度量,并采用误差平方和准则对时序运动点集进行运动分割。Assa 等提出了运动缩略图的概念,通过对输入运动数据的分析得到一个能够揭示运动主要内容的、可视化的缩略图。该方法的核心其实也是关键帧提取,使用 MDS 把运动数据降为低维“运动曲线”,然后通过迭代反复寻找“运动曲线”和平均曲线之间最大距离的点来确定关键姿态,也就是在全局意义上最与众不同的姿态。需要注意的是,一般的降维算法都不具有实时性,但是一旦计算出低维模型之后,新数据的合成和分析通常都能够实时得到。

7.5.3 分类在运动合成中的应用

分类算法通常能够实时地利用训练出的模型对输入数据进行预判,提高数据的使用效率。由于 Tang 等提出的连接动力学仿真和数据驱动过程的方法是用来模拟虚拟人受外力后的摔倒动作的,在连接动力学仿真和捕获数据时,需要对运动数据库进行搜索,这个过程比较费时。潘志庚等改进了 Tang 等的方法,他们利用神经网络(Neural Networks, NN)对捕获数据进行预分类,有效地减少了搜索时间。Sok 等提出了一种从运动捕获数据中学习控制策略的方法,首先修改捕获数据,使其尽量符合物理规律;然后从修改过的运动数据集中通过是一最近邻法(K-nearest Neighbor, KNN)对控制策略进行分类,这些控制策略可以让虚拟人鲁棒地模仿捕获数据的行为特征。在不影响结果的前提

下,能够对数据或者状态空间进行分类有效地提高运动合成的效率。

类似地,Arikan 等使用支持向量机对手工标注的运动片段进行学习,用于对运动序列自动标注,即自动对运动片段进行分类;Zhang 等在进行交互式拳击动画合成时使用了关系向量机——类似支持向量机的机器学习方法,来加速出拳规则和风格的创建;Peng 等也使用了一种简单的分类算法把相似帧聚在一起,起到运动分类的作用,以实现任意时刻在任意运动片段之间进行过渡;Tang 等提出了一个符合视觉感知规律的相似性度量方法,通过一种监督式的分类算法来定量地确定哪些度量得到的结果是最符合人的视觉规律的。

7.5.4 聚类在运动合成中的应用

聚类算法能够将相似的数据组织在一起,而这种相似关系通常可以得到运动数据和某些参数的依赖关系,对运动合成起到辅助作用。Ong 等使用一种层次化聚类模型,从捕获数据中学习符合人体运动规律的关节运动学约束知识和各个自由度之间的数据依赖关系,以如图 7-34 所示建立整个骨架的运动学约束模型,用来合成满足约束的人体动画。Lee 等 H 副提出的方法与运动图类似,将运动片段组织成图的结构。不同的是,该方法使用了一个两层模型,底层是通过 HMM 来组织的原始运动数据,高层是一个统计模型,将使用 k -均值(k -means)聚类后的相似帧汇聚成群,这些群之间是否可能产生过渡则通过概率来决定。

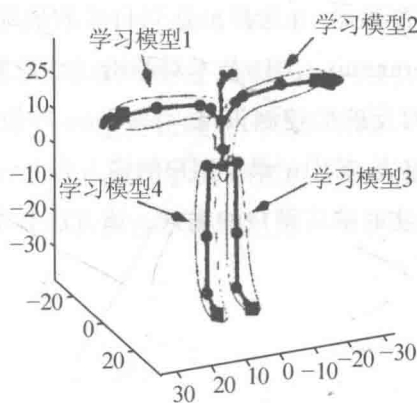


图 7-34 体运动的层次化聚类模型

在对运动数据处理,特别是在运动分割和检索中,聚类算法使用得很多。

Barbi 等使用 PCA 和概率 PCA 来分割运动数据,这是因为不同的运动行为应该有不同的内在结构,且在低维空间上表现为不同的聚类;该方法效果很好,但是运算效率较低。Sakamoto 等提出了运动图谱的概念,该方法首先使用 SOM 来对运动帧进行聚类 and 重组,构成运动图谱;然后用 SOM 节点来做索引,相似性度量也直接来自 SOM 的结果。然而他们的方法缺少对运动的高层索引,并且没有考虑运动的 timing,无法进行精确检索。Wu 等扩展了 Sakamoto 的方法,引入了 Smith-Waterman 序列相似性比较算法,加入叠层相似性度量,不仅考虑运动帧,还考虑运动片段。该方法不仅能够处理时间缩放问题,而且还在高层使用了简单的聚类,使得检索效率更高。

7.5.5 决策在运动合成中的应用

目前有一类方法是将切割成小片段的捕获数据根据用户的输入实时地连接起来,得到期望的运动。2002 年, Kovar 等提出了运动图的概念。将运动数据库构造成一个图的结构,图的节点是运动片段,有向边则代表片段之间可能的过渡,通过选择图中的某条通路得到一段较长的运动。为了能够更加有效地将运动片段连接起来,许多后续的工作都使用了机器学习对该方法进行扩展,决策方法就是其中之一,因为这种方法能够搜索出一个满足用户指定标准的最优的运动片段连接策略。增强学习(Reinforcement Learning, RL)技术是其中应用最广泛的,它原本应用于控制移动机器人。通过对训练时给出的延迟回报进行学习,获得一个控制策略,并选择能达到目的的最优动作。Lee 等使用动态规划(Dynamic Programming, DP)技术对拳击运动数据进行预处理,得到可以对外部环境进行实时反映的控制策略。McCann 等使用 RL 从用户的操作中训练出一张控制策略表,其中记录了用户的输入和下一段运动片段的对应关系,控制器根据此表来实时响应用户的输入。该方法非常适合交互式游戏,如图 7-35 所示。

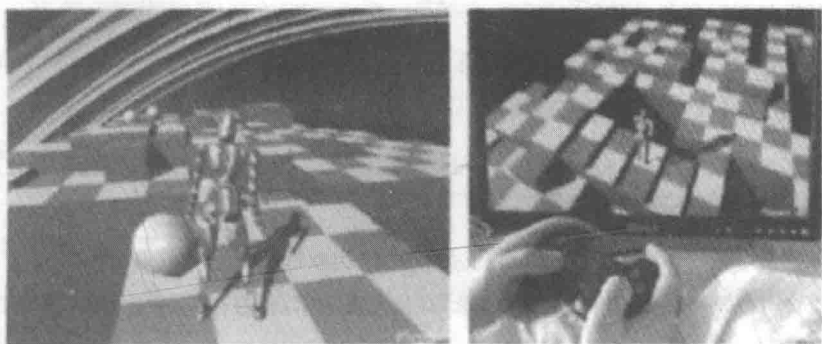


图 7-35 用 RL 实现基于片段的运动合成

Treuille 等的方法与 McCann 的方法类似,但其中简化了最优控制策略的学习,使用基函数插值来代替最优控制策略函数,实现次最优控制。Lo 等提出的方法与 McCann 的方法有以下 2 点不同:(1)使用了基于树的拟合迭代算法来学习控制策略,使算法更加鲁棒;(2)扩展了传统的 RL 框架,引入了参数化运动和插值方法,这样做的好处是在不影响控制效果的情况下可有效地减少输入样本的数量。Cheng 等也改进了 McCann 的方法,通过支持向量机的离线训练简化运动片段的搜索过程。Lee 等结合了 Treuille 和 Lo 的方法,使用参数化运动表示和 RL 实现复杂的运动控制。

为了达到这个目标,文献 E583 使用了两个技术:一是自动选择少量数据构建简单的运动控制器,并实现控制器之间的连接和转换;二是通过迭代逐步改进基函数,以实现高复杂度的控制策略函数;这样做就能对复杂行为进行控制。通过 RL 技术来学习控制策略是一种既耗时又耗内存的方法,如何针对具体应用简化 RL 过程是一个值得研究的问题。当然,一旦控制策略表生成,即可实现对虚拟人的实时控制。因此,对于那些希望对虚拟人进行实时和连续控制的应用,比如交互式游戏,这种学习控制策略的决策算法是一种有效的解决方案。

参考文献

- [1] SIMON R, MARK G. 机器学习基础教程[M]. 郭茂祖, 王春宇, 刘扬, 译. 北京: 机械工业出版社, 2014.
- [2] 李贤平. 概率论基础[M]. 北京: 高等教育出版社, 2010.
- [3] TOM M, MITCHEL L. 机器学习[M]. 曾华军, 张银奎, 等译. 北京: 机械工业出版社, 2003.
- [4] ETHEM A 机器学习导论[M]. 范明, 咎红英, 牛常勇, 译. 北京: 机械工业出版社, 2009.
- [5] 康威. 机器学习: 使用案例解析[M]. 陈开江, 刘逸哲, 孟晓楠, 译. 北京: 机械工业出版社, 2013.
- [6] IHN H, WITTEN E F. 数据挖掘—使用机器学习技术[M]. 董琳, 邱泉, 于晓峰, 译. 北京: 机械工业出版社, 2005.
- [7] 杨善林, 倪志伟. 机器学习与智能决策支持系统[M]. 北京: 科学出版社, 2004.
- [8] 马勇, 丁晓青. 基于层次型支持向量机的人脸检测[J]. 清华大学学报(自然科学版), 2003, 43(1): 35—38.
- [9] 叶航军, 白雪生, 徐光祐. 基于支持向量机的人脸姿态判定[J]. 清华大学学报(自然科学版), 2003, 43(1): 67—70.
- [10] 凌旭峰, 杨杰, 叶晨洲. 基于支撑向量机的人脸识别技术[J]. 红外与激光工程, 2001, 30(5): 318—322.
- [11] 张燕昆, 杨平, 刘重庆. 基于主元分析与支持向量机的人脸识别方法[J]. 上海交通大学学报, 2002, 36(6): 884—886.
- [12] 王宏漫. 欧宗瑛采用 PCA/ICA 特征和 SVM 分类的人脸识别[J]. 计算机辅助设计与图形学学报, 2003, 15(4): 416—420.
- [13] 忻栋, 杨莹春, 吴朝晖. 基于 SVM-HMM 混合模型的说话人确认[J]. 计算机辅助设计与图形学学报, 2002, 14(11): 1080—1082.
- [14] 柳回春, 马树元, 吴平东. UK 心理测试自动分析系统的手写体数字识别[J]. 北京理工大学学报, 2002, 22(5): 599—603.
- [15] 高学, 金连文, 尹俊勋. 一种基于支持向量机的手写汉字识别方法[J]. 电子学报, 2002, 30(5): 651—654.

- [16] 段立娟,崔国勤,高文.多层次特定类型图像过滤方法[J].计算机辅助设计与图形学学报,2002,14(5):1-6.
- [17] 庄越挺,刘骏伟,吴飞.基于支持向量机的视频字幕自动定位与提取[J].计算机辅助设计与图形学学报,2002,14(8):1-4.
- [18] 张磊,林福宗,张钊.基于支持向量机的相关反馈图像检索算法[J].清华大学学报(自然科学版),200242(1):80-83
- [19] 肖俊,吴飞,庄越挺.基于支持向量机与细节层次的三维地形识别与检索[J].计算机辅助设计与图形学学报,2003,15(4):410-415.
- [20] 陈光英,张千里,李星.基于SVM分类机的入侵检测系统[J].通信学报,2002,23(5):51-56.
- [21] 刘学军,陈松灿,彭宏京.基于支持向量机的计算机键盘用户身份验证[J].计算机研究与发展,2002,39(9):1082-1086.
- [22] 李晓黎,刘继敏,史忠植.基于支持向量机与无监督聚类相结合的中文网页分类器[J].计算机学报,2001,24(1):62-68.
- [23] 刘江华,陈佳品,程君.实用于人机交互的静态手势识别系统[J].红外与激光工程,2002,31(6):499-503.
- [24] 陈建华,包焯. Web 挖掘系统的设计与实现[J]. 计算机工程,2002,28(8):141-142.
- [25] 刘建伟,刘媛,罗雄麟.半监督学习方法[J].计算机学报,2015(8):1592-1617.
- [26] 周志华.基于分歧的半监督学习[J].自动化学报 2013,39(11):1871-1878.
- [27] 李凡长,何书萍,钱旭培.李群机器学习研究综述[J].计算机学报,2010,33(7):1115-1126.
- [28] 周志华.机器学习与数据挖掘[J].中国计算机学会通讯,2007.
- [29] 黄林军,张勇,郭冰榕.机器学习技术在数据挖掘中的商业应用[J].中山大学学报论丛,2005.
- [30] 施宇.基于数据挖掘和机器学习的木马检测系统设计与实现[D].成都:电子科技大学,2014.
- [31] 任昱衡,李倩星,米晓飞.数据挖掘—你必须知道的 32 个案例[M].北京:电子工业出版社,2016.
- [32] 朱文佳.基于机器学习的行人检测关键技术研究[D].上海:上海交通大学,2008.

- [33] 刘更代,潘志庚. 人体运动合成中的机器学习技术综述[J]. 计算机辅助设计与图形学报, 2010, 22(9): 1619-1627.
- [34] BELKIN M, NIYOGI P, SINDHWANI V. Manifold Regularization: A Geometric Framework for Learning from Labeled and Unlabeled Examples[J]. Journal of Machine Learning Research, 2006, 7(1): 2399-2434.
- [35] BLUM A, CHAWLA S. Learning from Labeled and Unlabeled Data using Graph Mincuts[C]// Eighteenth International Conference on Machine Learning. Morgan Kaufmann Publishers Inc. 2001: 19-26.
- [36] BLUM A. Combining labeled and unlabeled data with co-training[C]// Proceedings of the eleventh annual conference on Computational learning theory. ACM, 2000: 92-100.
- [37] CHAPELLE O, CHI M, ZIEN A. A continuation method for semi-supervised SVMs. [C]// International Conference. 2006: 185-192.
- [38] CHAPELLE O, B. Scholkopf, and A. Zien, eds. . Semi-Supervised Learning[J]. MIT Press, Cambridge, MA. 2006, 427(1): 533-549.
- [39] CHAPELLE O, WESTON J. Cluster Kernels for Semi-Supervised Learning[J]. Advances in Neural Information Processing Systems, 2003, 15: 15.
- [40] BOTTOU L, CHAPELLE O, DECOSTE D, et al. Trading Convexity for Scalability[C]// International Conference on Machine Learning. MIT Press, 2006: 201-208.
- [41] BREIMAN L. Bagging Predictors[J]. Machine Learning, 1996, 24(2): 123-140.
- [42] BREIMAN L. Stacked regressions[J]. Machine Learning, 1996, 24(1): 49-64.
- [43] BREIMAN L. Randomizing Outputs to Increase Prediction Accuracy[J]. Machine Learning, 2000, 40(3): 229-242.
- [44] BREIMAN L. Random forests[J]. Machine Learning, 2001, 45(1): 5-32.
- [45] BREIMAN L. Using Iterated Bagging to Debias Regressions[J]. Machine Learning, 2001, 45(3): 261-277.
- [46] FÜRNKRANZ J, GAMBERGER D, L N. Foundations of Rule Learning[M]. Heidelberg: Springer, 2012.
- [47] BRATKO I, MUGGLETON S. Applications of inductive logic programming [J]. 55 Communications of the ACM, 1995, 38(11): 65-70.
- [48] BRUNK C A, P M J. An investigation of noise tolerant relational concept learnin

gorithms[J]. In Proceedings of the 8th International Workshop on Machine Learning (IWML), 1991(8): 389-393.

[49] CARLSON A J, BETTERIDGE B, KISIEL B, et al. Toward an architecture for never ending language learning[J]. In Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI), 2010(24): 1306-1313.

[50] CENDROWSKA J. PRISM: An algorithm for inducing modular rules[J]. International Journal of Man-Machine Studies, 1987, 27(4): 349-370.

[51] COHEN W W. Fast effective rule induction, In Proceedings of the 12th International Conference on Machine Learning (ICML), 1995(12): 115-123.

[52] FIIRNKRANZ J. Top-down pruning in relational learning, In Proceedings of the 11th European Conference on Artificial Intelligence (ECAI), 1994(11): 453-457.

[53] FIIRNKRANZ J D, GAMBERGER, LAVRAC N. Foundations of Rule Learning [M]. Berlin: Springer, 2012.

[54] FIIRNKRANZ J, WIDMER G. Incremental reduced error pruning, In Proceedings of the 11th International Conference on Machine Learning (ICML), 1994(11): 70-77.

[55] GETOOR L, TASKAN B. Introduction to Statistical Relational Learning [J]. MIT Press, Cambridge, MA, 2007.

[56] JAEGER M. Relational Bayesian networks: A survey. Electronic Transactions on Artificial Intelligence, 2002, 6: 15.

[57] KERSTING K L, RAEDT D, KRAMER S. Interpreting Bayesian logic programs. In Proceedings of the AAAP 2000 Workshop on Learning Statistical Models from Relational, 2000(6): 29-35.

[58] LAVRAC N, DZEROSKI S. Inductive Logic Programming: Techniques and Applications. New York: Ellis Horwood, 2005.

[59] MICHALSKI R S. On the quasi-minimal solution of the general covering problem[J]. Symposium on Information Processing, 1969(50): 23-26.

[60] MICHALSKI R S. A theory and methodology of inductive learning, In Machine Learning: An Artificial Intelligence Approach, 1983(2): 111-161.

[61] MICHALSKI R S, MOZETIC I, HONG J, et al. The multi-purpose incremental learning system AQ15 and its test application to three medical domains[J]. In Proceedings of the 5th National Conference on Artificial Intelligence (AAAI), 1986(5): 1041-1045.

[62] MUGGLETON S. Inductive logic programming. *New Generation Computing*, 1991, 8(4): 295—318.

[63] MUGGLETON S. *Inductive Logic Programming*. Academic Press[M], London: UK, 1992.

[64] MUGGLETON S. Inverse entailment and Prolog. *New Generation Computing*, 1995, 13(3—4): 245—286.

[65] MUGGLETON S, BUNTINE W. Machine invention of first order predicates by inverting resolution[J]. In *Proceedings of the 5th International Workshop on Machine Learning (IWML)*, 1988(5): 339—352.

[66] MUGGLETON S, FENG C. Efficient induction of logic programs[J]. In *Proceedings of the 1st International Workshop on Algorithmic Learning Theory (ALT)*, 1990(1): 368—381.

[67] MUGGLETON S, LIN D. Meta-interpretive learning of higher-order dyadic datalog: Predicate invention revisited[J]. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI)*, 2013(23): 1551—1557.

[68] PLOTKIN G D. A note on inductive generalization, In *Machine Intelligence*, [M] Edinburgh: 1970.

[69] PLOTKIN G D. A further note on inductive generalization, In *Machine Intelligence* [M]. Edinburgh: Edinburgh University Press, 1971.

[70] RICHARDSON M, DOMINGOS P. Markov logic networks[J]. *Machine Learning*, 2006, 62(1—2): 107—136.

[71] ROBINSON J A. A machine-oriented logic based on the resolution principle[J]. *Journal of the ACM*, 1965, 12(1): 23—41.

[72] SRINIVASAN A. The Aleph manual. [EB/OL]. [1999-03-05]. <http://www.cs.ox.ac.uk/activities/machlearn/Aleph/aleph.html>.

[73] WNEK J, MICHALSKI R S. Hypothesis-Driven Constructive Induction in AQ17-HCI: A Method and experiments[J]. *Machine Learning*, 1994, 14(2): 139—168.

[74] SEUNG H S, LEE D D. Cognition[J]. *The manifold ways of perception*, 2000, 290(5500): 2268—2269.

[75] MITCHELL T M. *Machine Learning* [M]. New York: McGraw-Hill, 1997.

[76] BANDYOPADHYAY S, MAULIK U. *Knowledge Discovery and Data Mining*

- [M]. // Advanced Methods for Knowledge Discovery from Complex Data, 2004:3-42.
- [77] MJOLSNESS E, DECOSTE D. Machine learning for science: state of the art and future prospects[J]. Science, 2001, 293(5537): 2051-5.
- [78] MITCHELL, TOM M. Machine learning: an artificial intelligence approach[M]. Berlin: Springer, 1984.
- [79] CARBONELL J G. Machine learning: paradigms and methods[M]. Cambridge, MA: MIT Press, 1990.
- [80] AVRON, BARR, EOWARD A, et al. (The Handbook of Artificial Intelligence [M]. // The Handbook of artificial intelligence /. HeurisTech Press, 1981: 695-716.
- [81] MUGGLETON S. Inductive logic programming[J]. New Generation Computing, 1991, 8(4): 295-318.
- [82] RUMELHART D E, MCCLELLAND J L, PDP GROUP C. Parallel distributed processing: explorations in the microstructure of cognition, vol. 1: foundations[J]. Language, 1986, 63(4): 85.
- [83] VAPNIK V N. Statistical Learning Theory[J]. Encyclopedia of the Sciences of Learning, 2010, 41(4): 3185-3185.
- [84] DIETTERICH T G. Machine Learning Research: Four Current Directions[J]. Ai Magazine, 2000, 18(4): 97-136.
- [85] LO C P, YEUNG A K W. Concepts and Techniques of Geographic Information Systems (2nd Edition) (Ph Series in Geographic Information Science) [M]. // Concepts and techniques of geographic information systems /. State of New Jersey: Pearson Prentice Hall, 2007: 819-820.
- [86] ZHOU Z H. Three perspectives of data mining [J]. Artificial Intelligence, 2003, 143(1): 139-146.
- [87] WU B, NEVATIA R. Detection of Multiple [J]. Partially Occluded Humans in a Single Image by Bayesian Combination of Edgelet Part Detectors, 2005, 1: 90-97.
- [88] LAPTEV I. Improvements of Object Detection Using Boosted Histograms [C]// British Machine vision conference 2006, Edinburgh, UK, September. DBLP, 2006: 949-958. InProc. BMVC'06, Edinburgh, UK, pp. III: 949-958.
- [89] ZHU Q, AVADANS, CHEN M, et al. Fast Human Detection Using a Cascade of Histograms of Oriented Gradients [J]. IEEE Computer Society Conference on Computer Vi-

sion and Pattern Recognition, 2006(2):1491-1498.

[90] SABZME DANI P, MORI G. Detecting Pedestrians by Learning Shapelet Features [J]. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2007:1-8, 2007.

[91] TVZEL O, PORIKLI F, MEER P. Human Detection via Classification on Riemannian Manifolds [C]//Computer Vision and pattern Recognition, 2007. CVPR07. IEEE Conference on. IEEE, 2007:1-8.

书名

前言

目录

μ ú1?? D÷??

1.1 ê2? ' ê??ú?÷?S?°

1.2 ?ú?÷?S?° μ? · ¢?1à ú3ì?°?D??

??× '

1.2.1 ?ú?÷?S?° μ? · ¢?1à ú3ì

1.2.2 ?ú?÷?S?° μ??D????× '

1.3 ?ú?÷?S?° μ? · ?à à

1.3.1 ?ù ó ú?S?° 2???μ? · ?à à

1.3.2 ?ù ó ú?S?° · ? · " μ? · ?à à

1.3.3 ?ù ó ú?S?° · ?ê?μ? · ?à à

1.3.4 ?ù ó ú êy?YD?ê?μ? · ?à à

1.3.5 ?ù ó ú?S?° ??± ê μ? · ?à à

1.4 ?ú?÷?S?° μ?ó|ó??°?°

1.4.1 êy?Y · ???ó? í ú?ò

1.4.2 ?£ ê?ê?±e

1.4.3 ?ú é ú??D??¢?S é?μ?ó|ó?

1.4.4 ?ü1?à?μ?á ì ó ò

μ ú2?? ?ú?÷?S?°?ù ' ?à í??

2.1 òy??

2.2 ??D?? " ?£

2.2.1 ? " ò??£D í

2.2.2 ?£D í?ù é è

2.2.3 à í ???£D í μ?? " ò?

2.2.4 ×?D??t3??a

2.3 ?ò á?o í ???ó

2.4 ??D??£D í μ? · ???D?? ì ó |

2.5 · o? " ó?1y?ao?

2.5.1 ?é? □ êy?Y

2.5.2 ???é? □

2.5.3 K?????é? □ μ?????? · ?

2.6 ??é ù

2.7 ???ú ±?á?o í ???ê

2.7.1 ???ú ±?á?

2.7.2 ???ê o í ???ê · ???

2.7.3 ???ê μ??ó · "

- 2.7.4 ì??t???ê
- 2.7.5 áao????ê
- 2.7.6 ±??μ?´
- 2.7.7 ±´ò??1??ò
- 2.7.8 ?ú í??μ
- 2.8 3£??μ?à?éç·???
- 2.8.1 2???à?·???
- 2.8.2 ?t??·???
- 2.8.3 ?à??·???
- 2.9 á?D?D í???ú±?á??a???ê?ü?è
o´êy
- 2.10 3£??μ?á?D????ê?ü?èo´êy
- 2.10.1 ?ù?è?ü?èo´êy
- 2.10.2 |??ü?èo´êy
- 2.10.3 ?1?ü?èo´êy
- 2.10.4 ?à?a???1
- 2.11 ??è?1à??
- 2.11.1 êyY?´μ???è??μ
- 2.11.2 ×?´ó??è?
- 2.11.3 ×?´ó??è??aμ?ì?μ?
- 2.11.4 ×?´ó??è?·´êêó?óú?´?ó
- ?£D í
- 2.12 ??éù???êy1à??μ?ó°?ì
- 2.12.1 2?êy1à??μ?ó°?ì
- 2.12.2 2?êy1à??μ??è·?´D?
- 2.13 ?¤2a?μμ?±?òìD?
- 2.13.1 ?¤2a?μ±?òìD?ê?ày
- 2.13.2 1à???μμ??ú í??μ
- 2.14 ?à1à?ùéè
- 2.14.1 ?´?ú
- 2.14.2 1à???ùéè???è
- 2.14.3 2é?ùà í???ù´?
- 2.14.4 í?μ???D????μ?ò?°?·?
- ´
- 2.14.5 á????ùéè´í?ó?ê??μ???
- òì
- 2.14.6 ?§?°??·´±è??
- μú3??è?1¤é??-í???

3.1 ??ê?

3.1.1 ê2? ' ê?è?1 ▯ é??- í???

3.1.2 é??- í???μ??ù ±? ì?μ?ó?1

|?ü

3.2 é??- í???μ?ó|ó?á ì ó ò?é é ü

3.3 é ú??é??- í????ù ' ?

3.3.1 é ú??é??-?aμ??á11

3.3.2 é ú??é??-?aμ?D??ç ' |à í?

ú à í

3.4 è?1 ▯ é??- í????£D í? ° ??D??ü

3.4.1 è?1 ▯ é??-?a?£D í

3.4.2 è?1 ▯ é??- í????£D í

3.5 é??- í????§? °

3.5.1 Hebb?§? ° 1??ò

3.5.2 à?é ç?D?a?÷?§? ° 1??ò

3.5.3 á?D??D?a?÷?§? ° 1??ò

3.5.4 ×?D??ù · ??§? ° 1??ò

3.5.5 ?à1??§? ° 1??ò

3.5.6 ê ▯ ???a í ??§? ° 1??ò

3.5.7 í aD??§? ° 1??ò

3.6 é??- í????μ í3é è??ó?è í ó2?t

ê μ??

3.6.1 é??- í????μ í3× ü ì?é è??

3.6.2 é??- í???μ?è í?tê μ??

3.6.3 é??- í???μ??????a · ç? · ?

3

3.6.4 é??- í???μ?ó2?tê μ??

μ ú4?? ?÷ á ÷ μ??ú?÷?§? ° ?£D í

4.1 ê ÷?£D í

4.1.1 ????ê ÷

4.1.2 × ÷?a??D? · ??μ?ê ÷?§? ° ·

? · "

4.2 ???ê?£D í

4.2.1 ?y ì? · ???? ° ???o?ò aò?

4.2.2 ê?D?êy?Yμ????ê?£D í

4.2.3 í "1yó?? " ì ??t??è?ê μ??

?±eê??§? °

4.3 ?§3??ò á??ú

- 4.3.1 SVM $\mu \pm$
- 4.3.2 $\cdot D \ S \ \grave{a}$
- 4.3.3 $\ S \ \grave{a} \ \mu \ S \ \cdot$
- 4.3.4 $\ S \ \grave{a} \ \mu \ \acute{o} \ \acute{o} D$

$\times$

$\mu \ \acute{u} \ S \ \acute{u} \div \ S \ \cdot \cdot$

5.1 $\acute{u} \div \ S \ \mu \ \grave{a} \ \acute{u} \ \cdot$

5.2 $\ \grave{a} \ S \ \cdot$

5.2.1 $\ ' \ \pm \ \grave{e} \ \grave{u} \ \pm$

5.2.2 $\acute{e} \ \acute{u} \ \acute{e} \ \cdot \cdot$

5.2.3 $\ \acute{a} \ \acute{a}$

5.3 $\ ^{-} \ \acute{e} \ S \ \cdot$

5.3.1 $\ \grave{u} \ \acute{o} \ ^{-} \ \acute{e}$

5.3.2 Bagging $\acute{o} \ \acute{u} \ \acute{e} - \acute{a}$

5.4 $\ \grave{u} \ S \ \cdot$

5.4.1 $\ \grave{u} \ \pm$

5.4.2 $D \ \acute{a} \ S$

5.4.3 $\ |\ \acute{o} \ ^{-}$

5.4.4 $\ \acute{a} \ \acute{a}$

5.5 $\ \grave{u} \ \acute{o} \ \acute{u} \ S \ \grave{u} \ \mu \ ^{-} \ S \ \cdot$

5.5.1 $\ \grave{a} \ SVM$

5.5.2 $\ S \ \cdot$

5.5.3 SVM

5.6 $\ \grave{u} \ \acute{o} \ \acute{u} \ SVM \ \mu \ ^{-} \ S \ \cdot$

5.6.1 $\ \grave{u} \ \acute{o} \ \acute{u} \ SVM \ \mu \ S \ \acute{a} \ 11$

5.6.2 $\ \grave{u} \ \acute{o} \ \acute{u} \ 1 \ \grave{e} \ \pm \ ' \ \acute{u} \ \mu$

SVM

5.6.3 $\ \cdot \ 2 \ \grave{e}$

5.6.4 $\ \cdot \ D$

5.7 $\ \acute{a} \div D \ S \ \acute{o} \ \acute{í} \ \times \ S \ \cdot$

5.7.1 $\ \acute{a} \div D \ S \ \mu \ \grave{u} \ \pm$

5.7.2 $\ \acute{a} \div D \ S \ \mu \ \mu \ \cdot \cdot \ \grave{a}$

$\grave{a}$

5.7.3 $\ 11 \ ^{-} \ 1 \ \mu \ \acute{o} \ \mu \ \cdot \cdot$

5.8 $\ \grave{a} \ \grave{e} \ \acute{o} \ \acute{u} \div \ S \ \cdot$

5.8.1 $\ \grave{a} \ \grave{e} \ \acute{o} \ \acute{u} \div \ S \ \mu \ \grave{u} \ \pm$

$\ ?$

5.8.2 $\ \grave{a} \ \grave{e} \ \acute{o} \ \acute{u} \div \ S \ \mu \ \cdot \ \acute{o} \ ^{-} \ \grave{u} \ \acute{a}$

|?ù é è1?à í ??1?à í

5.8.3 à?èo?ú?÷?§?° μ??§?°?£D

í

μ ú6?? ' ó êy?Yê ± ' ú μ?êy?Y í ú?ò ó??

ú?÷?§?°

6.1 ????

6.2 ?ú?÷?§?° ó?êy?Y í ú?ò

6.2.1 ?T ' |2??ú μ??ú?÷?§?° ó?ê

y?Y í ú?ò

6.2.2 ?ú?÷?§?° ó?êy?Y í ú?ò ' ó .

ç ?1à ú3ì

6.3 êy?Y í ú?ò1□??

6.3.1 Weka

6.3.2 Javao í JVM ó?ò?

6.3.3 R ó???

6.3.4 Python

6.3.5 SAS

6.3.6 êy?Y í ú?ò1□??Wekao í R μ?± è

?? . ???

6.4 ?ù ó ú êy?Y í ú?òo í ?ú?÷?§?° μ?

??? í ? ì 2a?μ í 3

6.4.1 í ?ò3??? í ??ê?

6.4.2 í ?ò3??? í ? ì 2a?μ í 3Dè?ó .

???

6.4.3 í ?ò3??? í ? ì 2a?μ í 3é è??

6.4.4 ??? í ? ì 2a?μ í 3é è??D??á

6.5 ?ú?÷?§?° ??ê??ú êy?Y í ú?ò?D

μ?é ì ò μ ó | ó??D??

6.5.1 ??ê???? ì é ì ° ?ày?D??

6.5.2 ?ú??? . ??1ü à í μ?° ?ày?D?

?

6.5.3 ò?? - ó | ó ? í ???μ?° ?ày?D?

?

μ ú7?? ?ú?÷?§?° ê μ ó?° ?ày?a??

7.1 êy?Y . ???

7.1.1 . ???ó??é?□

7.1.2 ê2? ' ê?êy?Y

7.1.3 êy?Y í ???μ?à àD í

?

2????? ×